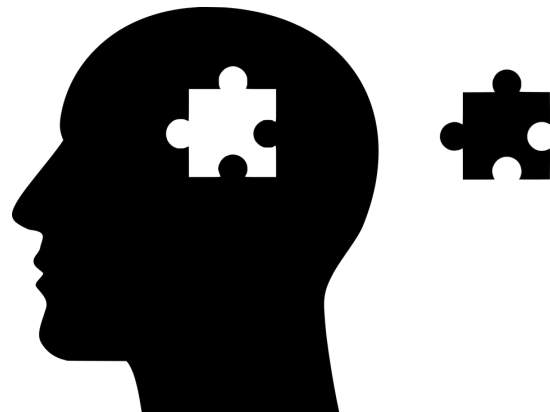


Lógica para Ciencias de la Computación

Laboratorio Prolog: Clase 1

Prolog

- Lenguaje de programación lógica orientado a la especificación de relaciones para responder consultas.
- Su nombre viene de “PROgrammation en LOGique”.
- En el contexto de la IA se refiere a las bases del conocimiento, haciendo énfasis en la complejidad de los datos y las deducciones que se pueden obtener de ellos.



¿Donde programaremos Prolog?

- Existen muchos IDEs que cumplen con todas las funcionalidades necesarias para realizar este curso.
- El software utilizado para obtener las capturas de estas transparencias es “SWI Prolog”.
 - Su página oficial es: <https://www.swi-prolog.org>



¿Cómo funciona el lenguaje?

- A diferencia de los lenguajes tradicionales, Prolog mantiene un diálogo continuo con el programador de inicio a fin.
- El sistema está creado de manera que el programador interroga al sistema Prolog.
- El sistema prolog inicia esperando una pregunta.

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.0.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.
```

```
For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
```

```
?- ■
```

Funcionamiento

- Lo escrito luego de “?-” es la pregunta que será formulada al sistema.
- Cada pregunta debe terminar con un punto.
- Al ejecutar la pregunta, el sistema mostrará en pantalla la respuesta correspondiente.

?- 20 is 15+5.

true.

?- 20 is 15+10.

false.

No olvidar

- Cada pregunta formulada a prolog debe terminar en un punto.
- Prolog considerará que la pregunta no está completa y seguirá esperando.

?- 3-2 is 1

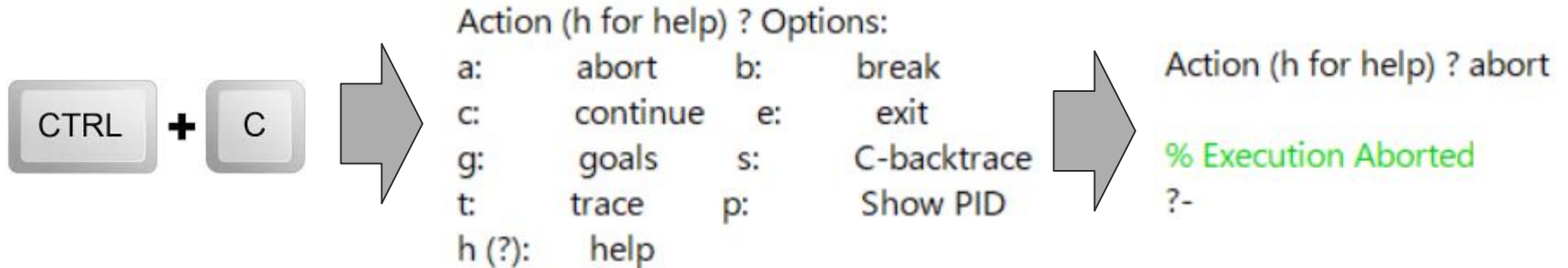
|
|
|
|
|

.

false.

¿Si quiero terminar una ejecución u obtener ayuda?

- Presionando CTRL+C activa las acciones de ayuda.



Sintaxis

- Al no cumplir con la sintaxis válida, Prolog indica dónde se encuentra el error que le impide al programa comprender la pregunta.

?- 5 is 7 2.

ERROR: Syntax error: Operator expected

ERROR: 5 is 7

ERROR: ** here **

ERROR: 2 .



?- 5 is 7-2.
true.

Conocimiento base de Prolog

- Al iniciar Prolog, conocimientos básicos, conceptos y definiciones de la aritmética de los números naturales con cargados.
- Sin embargo, Prolog desconoce la respuesta a preguntas que escapan de sus capacidades.
- Por ejemplo, “Philip es profesor?”:

?- esProfesor(Philip).

ERROR: Undefined procedure: esProfesor/1 (DWIM could not correct goal)

- Como Prolog no posee información sobre profesores, este indica que “**esProfesor**” no está definido. Entonces habrá que agregar este conocimiento a Prolog.

El conocimiento de Prolog

- Expresado mediante hechos y reglas, son representados utilizando casi exclusivamente lógica de primer orden.
- En consecuencia Prolog es un lenguaje lógico donde se puede realizar programación lógica.
- Un programa realizado en Prolog es un conjunto de hechos y reglas que expresan un conocimiento mediante lógica de primer orden.

Almacenar nuevo conocimiento

- Prolog almacena sus programas en ficheros con extensión '.pl'.
- Al leer estos programas, Prolog adquiere nuevo conocimiento.
- Prolog incluye predicados especiales útiles para navegar por el sistema de ficheros y visualizar los directorios.
- Un ejemplo de ruta de Prolog es C:/Program Files (x86)/Rutax (utiliza '/' en vez de '\\').

Predicados

- **pwd**: imprime el directorio actual.

```
?- pwd.
```

```
% c:/users/phili/onedrive/documentos/prolog/
```

```
true.
```

- **ls**: entrega la lista de ficheros de la dirección actual.

```
?- ls.
```

```
% Archivo1.pl  Archivo2.pl
```

```
true.
```

Modificar directorio de trabajo

- `cd`: modifica el directorio actual de trabajo de Prolog. El nombre del nuevo directorio debe ser una ruta, en notación Prolog y encerrada en comillas simples.

```
?- cd('c:/users/phili/onedrive/documentos/prolog/Carpeta2/').  
true.
```

Vamos a comenzar!



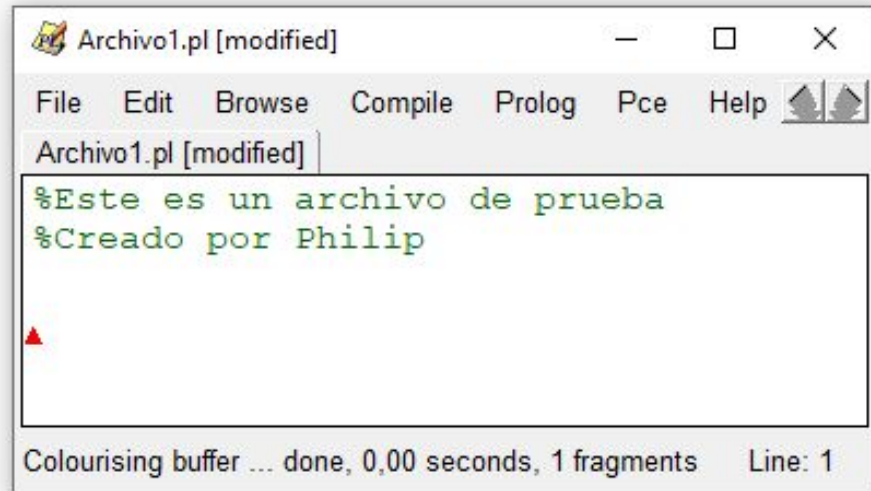
Predicados

- Expresan propiedades de los objetos (predicados monádicos) y otros además expresan relaciones entre ellos (predicados poliádicos).
- En prolog son llamados **Hechos**.
- Existen algunas condiciones a tener en cuenta:
 - Los nombres de los objetos y relaciones deben comenzar con una letra minúscula.
 - Se escribe primero la relación (predicado) y luego los argumentos separados por comas.
 - Al final de cada hecho debe ir un punto (“.”).

soy_un_predicado(argumento1, argumento2,...,argumenton) .

Vamos a crear nuestro primer archivo

- En él introduciremos todo nuestro conocimiento.
- Se recomienda comentar los códigos, orienta al revisor actual e incluso al creador mismo en el futuro.



The screenshot shows a window titled "Archivo1.pl [modified]" with a menu bar containing "File", "Edit", "Browse", "Compile", "Prolog", "Pce", and "Help". The text area contains two lines of Prolog code, both commented out with the % symbol. A red triangle cursor is positioned at the start of the first line. The status bar at the bottom indicates "Colourising buffer ... done, 0,00 seconds, 1 fragments" and "Line: 1".

```
%Este es un archivo de prueba  
%Creado por Philip
```

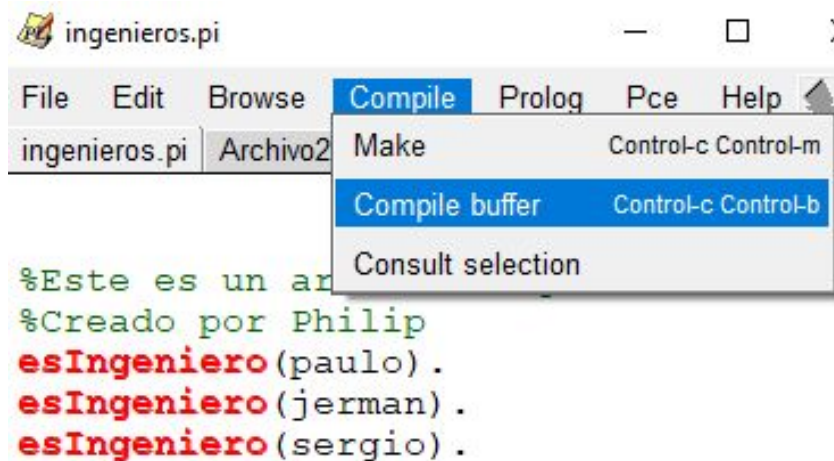
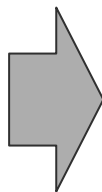
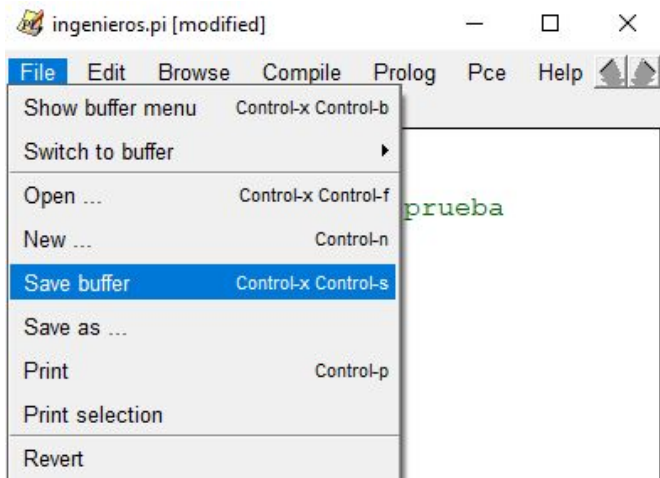

¿Qué conocimiento crearemos?

- Le enseñaremos a Prolog, quienes son ingenieros. Para esto se define un predicado que puede ser “esIngeniero(*variable*)”. *utilice minúscula
- El sistema nos responderá si *variable* es efectivamente un ingeniero.
- ¿Quienes son ingenieros?

```
%Este es un archivo de prueba  
%Creado por Philip  
esIngeniero(paulo) .  
esIngeniero(jerman) .  
esIngeniero(sergio) .
```

- Este conocimiento es una verdad incondicional (un hecho) y hay que almacenarlo en Prolog.

Almacenar conocimiento



% c:/users/phili/onedrive/documentos/prolog/ingenieros.pi compiled 0.02 sec, -1 clauses

Probemos la información que tenemos almacenada

- Hagamos algunas pruebas de lo conseguido.
- Comprobemos quienes son ingenieros.
- Estos predicados son **monádicos**, osea que son propiedades.

?- esIngeniero(francisco).

false.

?- esIngeniero(paulo).

true.

?- esIngeniero(jerman).

true.

?- esIngeniero(philip).

false.

Análisis de la respuesta

- Lo obtenido por el sistema dice que tanto Paulo como Jerman son ingenieros, mientras que Francisco y Philip no lo son.
- El conocimiento almacenado no contiene más información sobre los ingenieros, por lo tanto, es necesario agregar detalles que hagan su contenido más robusto.

Consulta I

- Si deseamos ver todas las personas que la base de conocimiento reconoce como ingeniero de debe utilizar el siguiente código:

```
?- esIngeniero(X).  
X = paulo |
```

- Observando lo entregado por el programa, se puede observar que el marcador de posición se queda esperando algo más. Con la tecla “Enter” el sistema considerará que la respuesta entregada satisface la consulta, si no es así, hay que escribir “;”. Lo que entregará una nueva respuesta.

Consulta II

- A continuación se ve el listado de elementos que son Ingenieros.

```
?- esIngeniero(X).
```

```
X = paulo ;
```

```
X = jerman ;
```

```
X = sergio.
```

Agregar nuevas condiciones

- Si queremos agregar nuevas condiciones a nuestro archivo debemos utilizar el comando **edit**(*nombreArchivo*).
- Abrirá el archivo indicado y se podrá editar.

```
?- edit(ingenieros).
```

```
Please select item to edit:
```

```
1 <loaded file>
```

```
'ingenieros.pi'
```

```
2 <loaded file>
```

```
'ingenieros.pl'
```

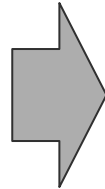
```
Your choice?
```

```
true.
```

A crear nueva información!

- Definimos que Philip y Bárbara son compañeros titulados de Jerman.
- Si preguntamos a Prolog, este nos mostrará si Philip o Francisco son compañeros de Jerman según lo que le hemos enseñado a la máquina.

```
esIngeniero(paulo).  
esIngeniero(jerman).  
esIngeniero(sergio).  
  
companioneroTitulado(philip,jerman).  
companioneroTitulado(barbara,jerman).
```



```
?- companioneroTitulado(philip,jerman).  
true.
```

```
?- companioneroTitulado(francisco,jerman).  
false.
```


Agregar condiciones

- Se agrega la condición que dice: “Una persona es ingeniero si se tituló con un compañero y su compañero es ingeniero”.
- Este es un predicado **poliádico**, osea que expresa relaciones.

```
esIngeniero(V) :- compTitulado(V,P), esIngeniero(P).  
esIngeniero(paulo).  
esIngeniero(jerman).  
esIngeniero(sergio).  
  
compTitulado(philip, jerman).  
compTitulado(barbara, jerman).
```



```
?- esIngeniero(X).  
X = paulo ;  
X = jerman ;  
X = sergio ;  
X = philip ;  
X = barbara ;
```

Actividad 1

- Cree una base de conocimiento que contenga a 3 de sus compañeros (incluido usted), asígnele un profesor y la facultad que pertenece el profesor.
- A la base de conocimiento del ejercicio anterior, agregue la información de edad y cree un hecho donde se pueda preguntar si el estudiante tiene o no 20 años.

Actividad 2

Cree una base de conocimiento en Prolog con las siguientes características:

- 3 alumnos participaron en el ramo “Introducción a la resolución de problemas de optimización aplicando metaheurísticas”, el resultado obtenido es el siguiente:
 - Caro : p1: 3.3 - p2: 4.5 - p3: 4.2
 - Michael: p1: 6.5 - p2: 2.7 - p3: 2.5
 - Gabriel: p1: 3.0 - p2: 5.5 - p3: 3.7
- Si cada nota tiene ponderación lineal, ¿Quiénes pasaron el ramo?, ¿Qué promedio tiene cada uno de los alumnos?

*Para cálculos puede utilizar **Var** **IS** **formula**, almacenará el cálculo en **Var**.

Actividad 3

- Generar una base de conocimiento que determine si un triángulo ingresado es:
 - Equilátero
 - Isósceles
 - Escaleno

Actividad 4

- Genere una base de conocimiento que contenga 10 elementos (cinco de carne y cinco de origen vegetal).
- Ingrese a la base de conocimiento la procedencia del elemento (vegetal o carne).
- Defina a los depredadores chupacabras (come carne) y chuparramas (come vegetales).
- Realice la consulta que indique todo lo que come el chupacabras y todo lo que come el chuparramas.