

Familiarización entorno Prolog

Curso 2003/04, Fecha:5/12/2003

Enunciado 1 (progenitor) *El siguiente fragmento incluye un conjunto de declaraciones básicas sobre una familia y una regla que indica si X es progenitor de Y*

```
progenitor(fernando ,juan ).
progenitor(luisa ,juan ).
progenitor(luisa ,maria ).
progenitor(luisa ,diego ).
progenitor(fernando ,pepe ).
progenitor(pepe ,pedro ).
progenitor(pedro ,ana ).
progenitor(pedro ,belen ).
```

```
hombre(juan ).
hombre(fernando ).
hombre(pepe ).
hombre(diego ).
hombre(pedro ).
```

```
mujer(luisa ).
mujer(pepa ).
mujer(belen ).
mujer(ana ).
```

Compile el programa y realizar las siguientes consultas:

- *Se cumple que fernando es el progenitor de pepe?*
- *Hay alguien que sea progenitor de ana?*
- *Hay alguien que sea progenitor de fernando?*
- *Hay alguien que sea a la vez progenitor de pepe y de juan?*

Enunciado 2 (padre) *El siguiente fragmento declara una regla que indica si X es el padre de Y*

$padre(X, Y) :- progenitor(X, Y), hombre(X).$

Declarar reglas familiares para:

- madre
- hermano
- hijo
- hermana

Enunciado 3 (abuelo) *El siguiente fragmento declara una regla que indica si X es abuelo de Y*

$abuelo(X, Y) :- progenitor(X, Z), progenitor(Z, Y), hombre(X).$

Declarar reglas para:

- abuela
- tío

Enunciado 4 (antepasado) *Definir un predicado que indique si X es un antepasado de Y*

Enunciado 5 (Ontología(Opcional)) *Una ontología consiste en una serie de definiciones de conceptos sobre un determinado dominio. Por ejemplo, las reglas anteriores definirían una ontología sobre relaciones familiares. Definir una ontología sobre algún otro dominio, como clasificaciones de animales, tipos de música, equipos de fútbol, etc.*

Enunciado 6 (Números naturales) *Una posible codificación de números naturales en Prolog se basa en la siguiente definición:*

- El cero es un número natural
- Si X es un número natural, entonces $s(X)$ (siguiente de X) también es un número natural

Las reglas anteriores se codificarían en Prolog como:

$natural(0).$
 $natural(s(X)) :- natural(X).$

- *Compile el programa anterior y comprobar si el dos ($s(s(0))$) es un número natural mediante la llamada *natural*($s(s(0))$)*
- *Observar las respuestas del sistema ante la pregunta *natural*(X)*

Enunciado 7 (suma) *La suma de números naturales puede definirse con las siguientes reglas.*

- $0 + X = 0$
- *Si $X + Y = Z$ entonces $s(X) + Y = s(Z)$*

En Prolog, dichas reglas pueden expresarse como:

suma($0, X, X$).
suma($s(X), Y, s(Z)$):- *suma*(X, Y, Z).

- *Realizar la pregunta *suma*($s(s(0)), s(s(0)), X$)*
- *Realizar la pregunta *suma*($s(s(0)), X, s(s(s(s(0))))$)*
- *Realizar la pregunta *suma*($X, Y, s(s(s(s(0))))$)*

Enunciado 8 (producto) *El producto de números naturales se define con las reglas:*

- $0 * X = 0$
- *Si $X * Y = Z$ entonces $s(X) * Y = Z + Y$*

Dichas reglas pueden expresarse en Prolog como:

producto($0, X, 0$).
producto($s(X), Y, R$):- *producto*(X, Y, Z), *suma*(Z, Y, R).

Realizar diversas consultas para comprobar que realmente multiplica naturales

Enunciado 9 (otros(Opcional)) *Intentar definir otros predicados aritméticos como:*

- *potencia*(X, Y, Z) *se cumple si Z es igual a X elevado a Y*
- *menor*(X, Y) *se cumple si X es menor que Y*
- *mcd*(X, Y, Z) *se cumple si Z es el máximo común divisor de X e Y*
- *etc.*