2 La aritmética en PROLOG

2.1 Variable instanciada

Se dice que una variable está *instanciada* si ha adquirido un valor. Se utiliza la palabra instanciar porque la asignación de un valor a una variable es temporal; Prolog sólo vincula valores a las variables mientras se está cumpliendo el objetivo, si éste falla la variable es desvinculada de ese valor.

En Prolog expresiones como $N = N1-2$ indican relaciones entre objetos. El valor de una expresión puede ser calculada si todas las variables están instanciadas en el momento de su evaluación. Si N no está instanciada el predicado igual se cumplirá vinculando a N el valor de la expresión de $N1 - 2$. $N1$ debe estar siempre instanciada a un valor ya que es parte de una expresión a evaluar.

2.2 El predicado igual

El predicado de igualdad está predefinido. Cuando se intenta satisfacer el objetivo

Goal: X=Y

Prolog intenta hacer coincidir X e Y, y el objetivo se satisface si ambas coinciden. Las reglas para decidir si X e Y son iguales son las siguientes:

- Si X es una variable no instanciada e Y está instanciada a un valor, entonces X e Y son iguales. Además, X quedará instanciada a lo que valga Y.
- Si X e Y son variables no instanciadas, el objetivo se satisface, y las dos variables quedan compartidas. Si dos variables quedan compartidas, en el momento en que una de ellas quede instanciada a un término, la otra queda automáticamente instanciada al mismo término.
- Las constantes son siempre iguales a sí mismas.
- Dos estructuras son iguales si tienen el mismo nombre y el mismo número de argumentos, y todos y cada uno de los correspondientes argumentos son iguales.

Como ya se ha indicado, una característica importante en Prolog y que lo diferencia de otros lenguajes de programación, es que una variable sólo puede tener un valor mientras se cumple el objetivo.

Hay dos predicados predefinidos que permiten determinar si una variable está instanciada o no. Son los predicados **free** y **bound**.

- **bound(X)**: Este predicado se cumple si la variable X está instanciada a algún valor.
- **free(X)**: Se cumple si la variable X no está instanciada a ningún valor.

2.3 Ejercicios propuestos

2.3.1 Definir una función

Programa en Prolog la siguiente función matemática

$$f(x) = \begin{cases} \text{no definida} & x < 0, \\ x & 0 \leq x < 3, \\ x - 3 & 3 \leq x < 6, \\ x - 6 & 6 \leq x < 9, \\ \text{no definida} & x \geq 9. \end{cases}$$

2.3.2 Las raíces de una ecuación de segundo grado

Dada la ecuación $ax^2 + bx + c = 0$ haz un programa en prolog que calcule sus raíces.

Nota: Prolog tiene un predicado predefinido, **sqrt(real)** que calcula la raíz de un número real.

2.3.3 El predicado suma

Escribe el predicado suma(X,Y,Z), de manera que Z es la suma de X e Y. Ejecuta

suma(2,3,8)
suma(2,3,Z)

¿Qué sucedería si preguntáramos **suma(X,3,8)**? Escribe un programa para que resuelva este problema y sepa responder al objetivo **suma(5,Y,8)**. Completa el programa para que resuelva **suma(X,Y,8)**, con $X, Y \geq 0$.

2.3.4 El predicado máximo

Escribe el predicado **maximo(X,Y,Z)**, de tal forma que Z sea el máximo de X e Y, donde la variable Z puede estar instanciada o no.

2.3.5 El mínimo común múltiplo

Escribe un predicado **mcm(X,Y,M)**, que sea capaz de calcular dados X e Y su mínimo común múltiplo M y además conteste *si/no* en el caso de que X,Y,M vengan instanciados.

2.3.6 El triángulo

Escribir un programa que dados tres valores a, b y c, determine si es posible construir un triángulo cuyos lados tengan longitud a, b, c. De ser así, indicar qué tipo de triángulo es: escaleno, isósceles o equilátero.

2.3.7 Los números naturales

Escribe un predicado **nat(X)** que sea capaz de generar los números naturales. Debe actuar de la siguiente forma:

goal: nat(6) yes
goal: nat(X) 12345678...

Escribe un predicado **mayorque(N,M)** que genere los números naturales mayores que uno dado, M.

2.3.8 El factorial

Escribe un predicado **fac(N,F)** que signifique que F es el factorial del número natural N para los siguientes casos:

(a) Dado un número natural N, que sea capaz de generar el factorial F.

(b) N y F vienen instanciados y debe contestar si/no.

(c) Ambas variables vienen sin instanciar y debe de generar todos los pares N, F.

Escribe un factorial universal que cubra todos los casos anteriores, es decir, que actúe correctamente ante toda entrada.