**PONTIFICIA UNIVERSIDAD CATOLICA MADRE Y MAESTRA**

**FACULTAD DE CIENCIAS DE LA INGENIERIA**

**INGENIERIA EN SISTEMAS COMPUTACIONALES**

**ISC-317 - Programación lógica**

***Práctica sobre COMPARACION DE TERMINOS.***

*Profesores: Antonio Rivero, José Moronta*

*Nombre: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Matrícula: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

**COMPARACION DE TERMINOS**

Lea la sección 9.1 del tutorial Learn Prolog Now en el siguiente enlace:

<http://lpn.swi-prolog.org/lpnpage.php?pagetype=html&pageid=lpn-htmlse37>

Al finalizar la lectura debe ser capaz de responder a las siguientes preguntas:

1. Identificar el uso de los operadores de comparación: = , \=, ==, \==, =:= , =\=

=The unification predicate.Succeeds if it can unify its arguments, fails otherwise.

\=The negation of the unification predicate.Succeeds if = fails, and vice-versa.

==The identity predicate.Succeeds if its arguments are identical, fails otherwise.

\==The negation of the identity predicate.Succeeds if == fails, and vice-versa.

=:=The arithmetic equality predicate.Succeeds if its arguments evaluate to the same integer.

=\=The arithmetic inequality predicate.Succeeds if its arguments evaluate to different integers.

1. Comparar listas correctamente.
2. Comprender el uso de los constructores de listas '[ ]' y '.' en Prolog.

**IDENTIFICACION DE TIPOS DE TERMINOS**

Lea la sección 9.3 del tutorial Learn Prolog Now en el siguiente enlace:

<http://lpn.swi-prolog.org/lpnpage.php?pagetype=html&pageid=lpn-htmlse39>

Al finalizar la lectura debe ser capaz de responder a las siguientes preguntas:

1. Identificar el uso de los predicados de verificación de tipos de términos:

|  |  |
| --- | --- |
| atom/1 |  |
| integer/1 |  |
| float/1 |  |
| number/1 |  |
| atomic/1 |  |
| var/1 |  |
| nonvar/1 |  |
| atom/1Is the argument an atom?  integer/1Is the argument an integer?  float/1Is the argument a floating point number?  number/1Is the argument an integer or a floating point number?  atomic/1Is the argument a constant?  var/1Is the argument an uninstantiated variable?  nonvar/1Is the argument an instantiated variable or another term that is not an un instantiated variable? |  |

1. Saber el uso de los predicados functor/3, arg/3, ‘=..’/2
2. Convertir átomos a cadenas con atom\_codes.

Responda a las siguientes preguntas:

1. Realice los ejercicios 9.1 a 9.4 del tutorial de Learn Prolog Now que están en el enlace:

<http://lpn.swi-prolog.org/lpnpage.php?pagetype=html&pageid=lpn-htmlse41>

1. Haga una tabla resumen de los predicados vistos en esta sección y su uso, como sigue:

|  |  |  |
| --- | --- | --- |
| Predicado (/n) | Uso(s) | Ejemplo(s) (Para cada uso pertinente escriba un ejemplo de consulta con el mismo y la respuesta que da). |
| =.. | Convierte un predicado y sus argumentos a una lista | '=..'(abc(ab, c),X). |
| Atom | Verifica si es un atomo | Atom(a).  True |
| Integer | Verifica si es un entero | Integer(5).  True |
| Float | Verifica si es un numero real | Float(3.2).  True |
| Number | Verifica si es un numero | Number(4).  True |
| Atomic | Verifica si es un atomo o numero | Atomic(a).  True |
| Var | Verificar que es una variable | Var(X).  True |
| Nonvar | Verifica que no es una variable | Nonvar(g).  True |
| Functor /3 | Nombre de la función y cantidad de argumentos | ?- functor(a(), N, A).  N = a, A = 0.  ?- functor(T, a, 0).  T = a. |
| Arg/3 | Acceder a un argumento en especifico | arg(1,fact(male(fred),23),F).  F = male(fred) |
| Atom\_codes | Convierte un atomo a una lista de código numéricos usados internamente para representar los caracteres del atomo | List = [112, 105, 122, 122, 97]   ?- atom\_codes(Atom, [98, 101, 101, 114]).  Atom = beer |

Suba a la PVA la tabla síntesis realizada con su nombre y matrícula.