Departamento de Cs. e Ingeniería de la Computación
Universidad Nacional del Sur

Inteligencia Artificial

Trabajo Práctico Nº 0

Repaso de Prolog

Segundo Cuatrimestre de 2009

Ejercicios

1. Dado el el siguiente programa Prolog

```
member(X,[X|Xs]).
member(X,[Y|Ys]):- member(X,Ys).

elim_rep([],[]).
elim_rep([X|Xs],Ys):- member(X,Xs),
                      elim_rep(Xs,Ys).
elim_rep([X|Xs],[X|Ys]):- elim_rep(Xs,Ys).
```

   a) Identificar: predicados, términos, variables, constantes, estructuras y functores.

   b) Determinar el resultado de las siguientes consultas:

```
?- elim_rep([2,1,tren,1], L).
?- elim_rep([ ],[a]).
?- member([a,b],[a,23+3,[a,b],hola(mundo)]).
?- member(X,[2+3+4]), X=9.
?- member(X,[1,2,3,4]), X = 5.
?- elim_rep([1,2|[3|[2]]],L).
```

2. Brindar las respuestas que daría Prolog frente a las siguientes consultas. Describir el significado intuitivo de cada consulta y su respectiva respuesta.

   a) `?- X = 3*4.`
   b) `?- X is 3*4.`
   c) `?- X-5 = 8-Y.`
   d) `?- X + 3 is 2 + 3.`
   e) `?- 7 \= 9.`
   f) `?- X \= 9.`
   g) `?- X-5 \= 8-Y.`
   h) `?- 5-X \= 8-Y.`

3. Un conjunto puede ser modelado mediante una lista de elementos sin repeticiones. Adoptando esta representación, implementar en Prolog las siguientes operaciones sobre conjuntos.

   a) Comprobar si una lista de elementos constituye un conjunto válido.

   b) Determinar si un elemento pertenece a un conjunto.

   c) Incorporar un elemento a un conjunto.

   d) Unir dos conjuntos.

e) Intersectar dos conjuntos

f) Calcular la diferencia entre dos conjuntos.

g) Dada una lista de elementos con repeticiones, construir un conjunto que contenga todos los elementos de esa lista.

4. Sean consideradas como válidas las expresiones descriptas por la siguiente BNF:

   $E \quad ::= \quad E + E \quad | \quad E * E \quad | \quad E - E \quad | \quad (E) \quad | \quad C$

   $C \quad ::= \quad$ "una lista representado a un conjunto"

   Asumiendo que $+$, $*$ y $-$ denotan las operaciones sobre conjuntos de unión, intersección y diferencia respectivamente, definir en Prolog un predicado eval/2 que tome una expresión y retorne el resultado correspondiente a su evaluación. Por ejemplo:

   ?- eval(([z, t] * [t, u]) + [x, y], Rta).

   Rta = [t, x, y]

   ?- eval([a,b,c] + ([h,i,j,k,l] * ([j,k,l,m,n,t,u] - [m,n])), Rta).

   Rta = [a,b,c,j,k,l]

5. Definir un predicado Prolog `elim_cant/4` que dada una lista L y un elemento E retorne:

   a) La lista L sin el elemento E.

   b) La cantidad de veces que se borró E de L.

   Por ejemplo:

   ?- elim_cant([a,b,c,d,e,a,b,c,d,a,b,c,a,b,a], b, LsinE, Cant).

   LsinE = [a,c,d,e,a,c,d,a,c,a,a]

   Cant = 4

6. Definir un predicado Prolog `reconocer/1` que al recibir como entrada una lista que representa una cadena de símbolos terminales determine si la misma pertenece al lenguaje $L = \{\lambda\} \cup \{a^n b^{2n} c^n \mid n \geq 1\}$, donde λ representa la cadena vacía.

   Por ejemplo:

   ?- reconocer([a,a,b,b,b,b,c,c]).
   yes.

   ?- reconocer([a,a,b,b,b,c]).
   no.

7. Definir un predicado Prolog `prod_cart/3` que reciba como argumentos dos conjuntos $C1$ y $C2$ (representados mediante listas) y retorne $C1 \times C2$, el producto cartesiano entre $C1$ y $C2$. Por ejemplo:

   ?- prod_cart([a,b,c], [1,2], X).

   X = [[a,1],[a,2],[b,1],[b,2],[c,1],[c,2]]

   Considerar la resolución de este ejercicio utilizando el predicado `findall/3`.

8. Definir un predicado PROLOG `prod_cart/2` que reciba como argumento una lista de conjuntos $[C1, C2, ..., Cn]$ y retorne el conjunto correspondiente al producto cartesiano $C1 \times C2 \times ... \times Cn$. Por ejemplo:

   ?- prod_cart([[1,2,3], [a,b], [x,y]], PC).

   PC = [[1,a,x], [1,a,y], [1,b,x], [1,b,y], [2,a,x], [2,a,y], [2,b,x], [2,b,y], [3,a,x], [3,a,y], [3,b,x], [3,b,y]]

   Ayuda: Tratar de resolver este ejercicio extendiendo la solución encontrada para el ejercicio anterior (aquella que utiliza el predicado `findall/3`).

9. Implementar en PROLOG la versión recursiva y la versión iterativa de las siguientes funciones:

   a) Fibonacci

   b) Factorial

   c) Coeficientes binomiales

10. Definir un predicado PROLOG `aplanar/2` que relacione una lista de elementos con su imagen aplanada. Por ejemplo:

    ?- aplanar([a,[a,b],[1,2,[a],[]]], X).

    X =[a,a,b,1,2,a].

11. Analizar la semántica de los siguientes predicados:

```
uno([X|Xs],Ys,[X|Rs]):- member(X,Ys), uno(Xs,Ys,Rs).
uno([X|Xs],Ys,Rs):- not member(X,Ys), uno(Xs,Ys,Rs).
uno([],Ys,[]).

dos([X|Xs],Ys,Rs):- member(X,Ys), dos(Xs,Ys,Rs).
dos([X|Xs],Ys,[X|Rs]):- not member(X,Ys), dos(Xs,Ys,Rs).
dos([],Ys,Ys).
```

12. Escribir predicados PROLOG que implementen los siguientes métodos de ordenamiento para listas de enteros:

    a) *InsertSort*

    b) *SelectSort*

    c) *MergeSort*

    d) *QuickSort*

13. Un *orden* puede representarse en PROLOG mediante un predicado de aridad 2, tal que dados dos elementos retorne `yes` si éstos respetan el orden especificado y `no` en caso contrario. Implementar un predicado PROLOG `ordenarGen/3` (ordenar genérico) que reciba como argumentos una lista de elementos y el nombre de un predicado implementando un orden y retorne la imagen ordenada de la lista de acuerdo al orden suministrado.

Como se muestra a continuación, podríamos utilizar `ordenarGen/3`, por ejemplo, para ordenar lexicográficamente (alfabéticamente) una lista de palabras, donde una palabra está representada mediante una lista de letras y `lexicograf/2` es un predicado que implementa el orden lexicográfico.

```
?- ordenarGen([[c,a,s,a],[p,e,r,r,o],[a,u,t,o],[a,u,l,a]],lexicograf, L).

L= [[a,u,l,a], [a,u,t,o], [c,a,s,a], [p,e,r,r,o]]
```

14. Analizar el efecto del cut (`!`) empleado en el siguiente programa Prolog. Determinar qué respuestas se obtienen de la consulta `?- p(X,Y)`. Explicar claramente por qué se obtienen esas respuestas.

```
p(X,Y) :- q(X), !, r(Y).
p(c, 1).
q(a).
q(b).
r(1).
r(2).
```

15. Considerando las siguientes tres representaciones para digrafos (grafos dirigidos), defina un predicado `arco(U,V)` que determine si el arco (u, v) pertenece a un digrafo.

    a) El digrafo es una colección de hechos de la forma `arc(U,V)` indicando la existencia de un arco del nodo `U` al nodo `V`.

    b) El digrafo es una colección de hechos `adyacentes(N,L)` que modelan la relación de adyacencia, donde `N` es un nodo y `L` la lista de nodos adyacentes a `N`.

    c) El digrafo se representa mediante un hecho `digrafo(G)`, donde `G` es una lista de pares `N-L`, donde a su vez `N` es un nodo y `L` la lista de nodos adyacentes a `N`.