REPRESENTATION OF LIST IN PROLOG PROGRAMMING

LIST

What is a List?

The list is a simple data structure widely used in non-numeric programming. List consists of any number of items, such as red, green, blue, white, dark. In Prolog it will be represented as:

[red, green, blue, white, dark]

A list can be either empty or non-empty. In the first case, the list is simply written as a Prolog atom, []. In the second case, the list can be viewed as consisting of two things:

(1) the first item, called the **head** of the list;
(2) the remaining part of the list, called the **tail**.

EXAMPLE

What is a List? (Contd.)

For our example list

[red, green, blue, white, dark]

the head is red and the tail is the list

[green, blue, white, dark]

Let us consider L = [a, b, c]

If we write Tail = [b, c] then we can also write L = [a | Tail]

Here the vertical bar separates the head and tail.

So following list representations are also valid

[a, b, c] = [x | [b, c]] = [a, b | [c]] = [a, b, c | [ ]]

As a definition we can say –

A list is a data structure that is either empty or consists of two parts: a head and a tail. The tail itself has to be a list.

MEMBER OF A LIST

To check whether an object X is member of list L or not.

list_member(X, L).
where X is an object and L is a list.

The goal list_member(X, L) is true if X occurs in L.

For example,

list_member(b, [a, b, c])	is true,
list_member(b, [a,[b, c]])	is not true, but
list_member ([b, c] [a,[b, c]])	is true.

The program for the membership relation can be based on the following observation:

X is a member of L if either
(1) X is the head of L, or
(2) X is a member of the tail of L.

Let us go for a demonstration...

MEMBERSHIP FUNCTION

```
% file: C:/Arnab Docs/PROLOG/PrologWorkSpace/list1.pl

list_member(A, [A|_]).
list_member(A, [_|B]) :-
        list_member(A, B).

(16 ms) yes
| ?- list_member(a,[a,b,c,d,e,f]).
```

PROGRAMS

Hints:

Question	Program Code	Input	Output
Insert an element in a list.	insert(X,L,[X|L]).	insert(k,[a],L).	L = [k, a].
		insert(5,[a,r,3],L).	L = [5, a, r, 3].
Insert an element in a	insert(X,[],[X]).	insert(k,[a],L).	L = [a,k].

PROGRAMS

last position of a list.	insert(X,[Y|Tail],[Y|Tail1]):- insert(X,Tail,Tail1).	insert(5,[a,r,3],L).	L = [a, r, 3,5].
Delete an element in a list.	del(X,[X|Tail],Tail). del(X,[Y|Tail],[Y|Tail1]):- del(X,Tail,Tail1).	del(a,[a,b,a,a],L).	L = [b, a, a] ; L = [a, b, a] ; L = [a, b, a] ; false.
		del(6,[4,6,7],L).	L = [4, 7] ; false.
Generate all permutations of a list of elements.	del(X,[X|Tail],Tail). del(X,[Y|Tail],[Y|Tail1]):- del(X,Tail,Tail1). permu([],[]). permu([X|L],P):- permu(L,L1),del(X,P,L1).	permu([red,green,blue],P).	P = [red, green, blue] ; P = [green, red, blue] ; P = [green, blue, red] ; P = [red, blue, green] ; P = [blue, red, green] ; P = [blue, green, red] ; false.