**PROBLEM STATEMENT**:

1. Write a goal to delete the first 3 elements and the last 3 elements from a list L producing list L1.
2. Write a goal using conc. to delete the last 3 elements from a list L producing another list L2.
3. Define the relations : last (Item list) so that Item is the last element of a link list

Write 2 versions:

1. Using conc. Relation
2. Without using conc. relation .
3. Define 2 predicates:

Evenlength (List) and oddlength (List) so that they are true if their argument is a list of even or odd lengths respectively.

1. Define the relation shift(List1,List2) such that List2 is List1 “shifted rotationally" by 1 element to the left.

**CODE**

1. conc([],L,L)

conc([x],[L1],L2[x],[L3]):- conc(L1,L2,L3)

del(L,L1.L2):- conc([\_,\_,\_]L1,L)conc(L1 [\_,\_,\_],L2)

?-del([1,2,3,4,5,6,7],L1,L2) write (L2)

1. conc([],L,L)

conc([x][L1],L2[x],[L3]):- conc(L1,L2,L3)

?- conc(x,[\_,\_,\_],[1,2,3,4,5,6,7,8]) write (x)

1. using conc.

conc ([],L,L)

conc([x/L1], L2, [x/L3]):- conc(L1,L2,L3)

last(x,L1):-conc(L,[x],L1)

?-last(x,[1,2,3,4,5,6,7]) write(x)

1. Without using conc.

last (x,[x]).

last (x,[HIT]):- last(x,T)

?- last(x,[1,2,3,4]) write (x)

1. evenlength([])

evenlength(HIT):- oddlength (T)

oddlength ([])

oddlength(HIT):- evenlength (T)

?- evenlength([1,2,3,4,5]).

?- oddlength([1,2,3]).

1. conc([],L,L)

conc([x/L1], L2, [x/L3]):- conc(L1,L2,L3).

Shift ([First/Rest], shifted):- conc(Rest, First, shifted)

?- shift([1,2,3,4,5], List) write (List)

**OUTPUT**

1. [4] Yes

No

1. [1,2,3,4,5] Yes

No

1. [5] Yes

No

1. [4] Yes

No

1. Yes

Yes

No

1. [2,3,4,5,1] Yes