ASSIGNMENT - 2

ANIKEIT SETHI(190001003)

**Q1)**

**analyse\_list(L) :-**

**L = [H | T] ,**

**format('This is the head of your list:~w~n',[H]), format('This is the tail of your list:~w~n',[T]).**

**analyse\_list([]) :-**

% input is an empty list

**write('This is an empty list'),!.**

**analyse\_list(X) :-**

% if input is not list, prevent backtracking, fail.

**\+ is\_list(X),!,1=0.**

**Q2.)**

**remove\_duplicates([], []).** % empty list input

**remove\_duplicates([H | T], Result) :-**

% Head is a member of Tail => H is repeating => remove head from list **member(H, T),!,**

**remove\_duplicates(T, Result)** % use cut to prevent backtracking

**remove\_duplicates([H | T], [H | Result]) :-**

% As Head is not a member of Tail => Add Head to List and call remove\_duplicates with Tail and updated List **remove\_duplicates(T, Result).**

**Q3.)**

% only 2 elements are present in the list, assign first one to X **last\_but\_one([Y,\_], Y).**

%first case fails=>recurse the function over tail of the list **last\_but\_one([\_|w], y) :**

**last\_but\_one(W, y).**

**Q4.)**

**kth\_element(X, [X|\_], 1).**

**kth\_element(X, [\_|T], K) :- kth\_element(X, T, Knew), Knew is K - 1.**

**Q5.)**

**rev([],[]).**%Empty

**rev([R],[R]).**%single element

**rev([H|T],R):- rev(T,TEMP), append(TEMP,[H],R).**

**Q6.)**

**pali([]).** % empty list is palindrome

**pali([\_]).** % single element list is also palindrome

**pali(L) :-**

% H|Rem assigns H = first element, Rem = 2nd to last element

% [H] => last element

% if H from both the statements is same => Check if Rem is

palindrome or not **append([H|Rem], [H], L), pali(Rem).**