Practical 4 : Write a program to perform following operations of List:

i. to display elements of list

ii. to check given element is in the list or not

iii. to print last element of the list

iv. to print sum of elements of given list

Objective:

To implement basic list operations in Prolog, including displaying list elements, checking membership, finding the last element, and calculating the sum of elements.

Description:

This Prolog program defines a set of basic list operations using recursive predicates. The display_list/1 predicate is used to traverse a list and display each element on a new line, demonstrating how recursion can be applied to process lists. The is_member/2 predicate checks whether a given element exists in a list, functioning similarly to the built-in member/2 predicate, and uses pattern matching to efficiently identify the element. The last_element/2 predicate finds the last element of a list by recursively traversing to the end, where a single element list is reached. Finally, the sum_list/2 predicate calculates the sum of numeric elements in a list by recursively adding the head of the list to the sum of the tail. Together, these predicates illustrate fundamental Prolog concepts such as recursion, backtracking, and list pattern matching, and they serve as foundational tools for more complex logical reasoning and data manipulation tasks.

Code:

```
% i) Display all elements of a list
display_list([]).
display_list([H|T]) :-
    write(H), nl,
    display_list(T).

% ii) Check if an element is in the list
is_member(X, [X|_]) :- !.
is_member(X, [_|T]) :-
    is_member(X, T).

% iii) Print the last element of the list
last_element([X], X).
last_element([_|T], X) :-
    last_element(T, X).

% iv) Sum of all elements in the list
```

```
sum_list([], 0).
sum_list([H|T], Sum) :-
   sum_list(T, Rest),
   Sum is H + Rest.
```

Output: