```prolog
%calculate the length
len([],0).
len([_|T],N) :- len(T,X), N is X+1.

%check the length
isEmpty(L) :- len(L,X),X=:=0.

%P1
max_of_two(X,Y,X) :- X >= Y, !.
max_of_two(X,Y,Y) :- X < Y.
list_max_elem([X],X).
list_max_elem([X|Rest],Max) :-
   list_max_elem(Rest,MaxRest),
   max_of_two(X,MaxRest,Max).

%P2
list_sum([],0).
list_sum([Head|Tail], Sum) :-
   list_sum(Tail,SumTemp),
   Sum is Head + SumTemp.


%P3
evenlength([]).
evenlength([H1|[H2|T]]):-evenlength(T).

oddlength(L):- \+ evenlength(L).

%P6
appendList([],L,L).
appendList([H|T],L2,[H|L3])  :-  appendList(T,L2,L3).

%P4
naiverev([],[]).
naiverev([H|T],R):-  naiverev(T, RevT),  appendList(RevT, [H], R).

accRev([],A,A).
accRev([H|T],A,R):-  accRev(T,[H|A],R).
revAcc(L,R):-  accRev(L,[],R).

%P5
is_palindrome(L) :- revAcc(L,L).

%P7
remove([], [], X).
remove([X|T1], L, X) :- remove(T1, L, X), !.
remove([H|T1], [H|T2], X) :- remove(T1, T2, X).

%P9
remove_nth([H|T], 1, T).
remove_nth([H|T1], N, [H|T2]) :- N > 1,
   N1 is N - 1, remove_nth(T1, N1, T2).

%P8
insert_nth(X, 1, L, [X|L]).
insert_nth(X, N, [H|T1], [H|T2]):- N > 1, N1 is N - 1, insert_nth(X, N1, T1, T2).

%P10
diffList([], [], 0).
diffList([H|T1], [H|T2], X):-  diffList(T1, T2, X), !.
diffList([H1|T1], [H2|T2], X):-  diffList(T1, T2, X1), X is X1 + 1.

isSortedList([_|[]]).
isSortedList([H1|[H2|T]]):-H2>=H1, isSortedList([H2|T]).

countOccurrenceList([], _, 0).
countOccurrenceList([H|T], X, N):- H =:= X, countOccurrenceList(T, X, N1), N is N1+1.
countOccurrenceList([H|T], X, N):- H =\= X, countOccurrenceList(T, X, N1), N is N1.

searchIndexList([], _, 0).
searchIndexList([H|_], H, 1).
searchIndexList([H|T], X, N):- H =\= X,
    searchIndexList(T, X, N1), N1 =\= 0, N is N1+1.


```