

# ASSIGNMENT ARTIFICIAL INTELLIGENCE

## RAMANUJAN COLLEGE



UNIVERSITY OF DELHI

---

### SUBMITTED BY :

NAME : Aryan Yadav  
ROLL NO. : 24570012  
EXAMINATION ROLL NO :  
24020570016  
CLASS : BSC(H) COMPUTER  
SCIENCE  
SEM- 3

---

---

### SUBMITTED TO:

Mrs. Bhavya Ahuja  
ASSISTANT PROFESSOR  
RAMANUJAN COLLEGE,  
UNIVERSITY OF DELHI

---

1. Write a PROLOG program to implement the family tree and demonstrate the family relationship.

```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl
% ----- Facts -----

% Gender
male(john).
male(mike).
male(ryan).
female(lisa).
female(emma).

parent(john, mike).
parent(lisa, mike).
parent(mike, ryan).
parent(emma, ryan).

% Rules
father(F, C) :- male(F), parent(F, C).
mother(M, C) :- female(M), parent(M, C).
grandparent(GP, GC) :- parent(GP, P), parent(P, GC).
sibling(A, B) :- parent(P, A), parent(P, B), A \= B.
brother(B, X) :- sibling(B, X), male(B).
sister(S, X) :- sibling(S, X), female(S).

% Ancestor (recursive)
ancestor(A, X) :-
    parent(A, X).
ancestor(A, X) :-
    parent(P, X),
    ancestor(A, P).
▲
```

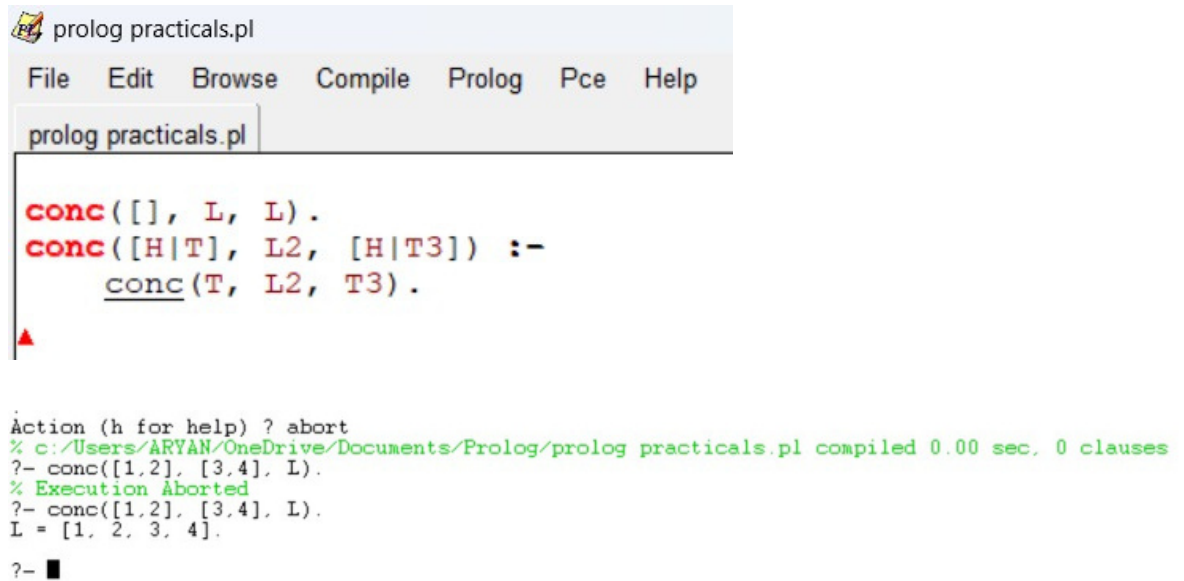
Output:

```
?- father(F, mike).
F = john ,

?- mother(M, ryan).
M = emma.

?- sibling(mike, emma).
false.
```

2. Write a PROLOG program to implement `conc(L1, L2, L3)` where L2 is the list to be appended with L1 to get the resulted list L3.

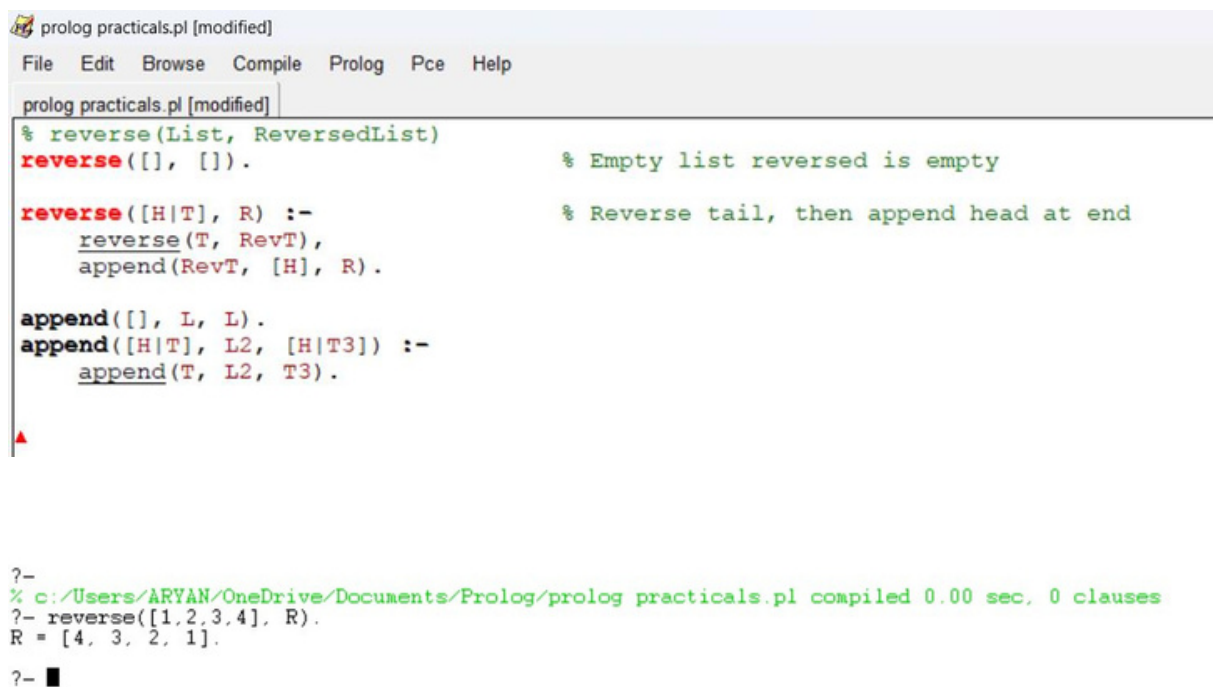


```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl

conc([], L, L).
conc([H|T], L2, [H|T3]) :-
    conc(T, L2, T3).

Action (h for help) ? abort
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog practicals.pl compiled 0.00 sec, 0 clauses
?- conc([1,2], [3,4], L).
% Execution Aborted
?- conc([1,2], [3,4], L).
L = [1, 2, 3, 4].
?-
```

3. Write a PROLOG program to implement `reverse(L, R)` where List L is original and List R is reversed list.



```
prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]

% reverse(List, ReversedList)
reverse([], []). % Empty list reversed is empty

reverse([H|T], R) :- % Reverse tail, then append head at end
    reverse(T, RevT),
    append(RevT, [H], R).

append([], L, L).
append([H|T], L2, [H|T3]) :-
    append(T, L2, T3).

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog practicals.pl compiled 0.00 sec, 0 clauses
?- reverse([1,2,3,4], R).
R = [4, 3, 2, 1].
?-
```

4. Write a PROLOG program to calculate the sum of two numbers.

```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl
sum(A, B, Z) :-
    Z is A + B.
▲

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
?- sum(5, 7, R).
R = 12.
-
```

5. Write a PROLOG program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y.

```
prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]
max(X, Y, X) :- X >= Y.
max(X, Y, Y) :- Y > X.
▲

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
?- max(10, 7, M).
M = 10
```

6. Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.

```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl
factorial(0, 1).
factorial(N, F) :-
    N > 0,
    N1 is N - 1,
    factorial(N1, F1),
    F is N * F1.
▲

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
?- factorial(5, F).
F = 120
```

7. Write a program in PROLOG to implement generate\_fib(N,T) where T represents the Nth term of the Fibonacci series.

```
prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]

% Base cases
fib(0, 0).
fib(1, 1).

% Recursive case
fib(N, T) :-
    N > 1,
    N1 is N - 1,
    N2 is N - 2,
    fib(N1, T1),
    fib(N2, T2),
    T is T1 + T2.

Warning: Redefined static procedure fib/2
Warning: Previously defined at c:/users/aryan/one
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog p
ib(6, T).
Unknown action: f (h for help)
Action? .

?- fib(6, T).
T = 8
```

8. Write a PROLOG program to implement power (Num, Pow, Ans):where Num is raised to the power Pow to get Ans.

```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl

power(_, 0, 1). % Anything
power(Num, Pow, Ans) :-
    Pow > 0,
    P1 is Pow - 1,
    power(Num, P1, A1),
    Ans is Num * A1.

Warning: Redefined static procedure power/3
Warning: Previously defined at c:/users/aryan/one
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
over(2, 5, A).[print]
T = 8 .

?- power(2, 5, A).
A = 32
```



9. PROLOG program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.

```

prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]
multi(N1, N2, R) :-
    R is N1 * N2.
▲
A = 32
% c:/Users/ARYAN/OneDrive/Documents/Prolog/p
ulti(6, 7, R).
Unknown action: m (h for help)
Action? .
?- multi(6, 7, R).
R = 42.

```

10. Write a PROLOG program to implement memb(X, L): to check whether X is a member of L or not.

```

prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]
memb(X, [X|_]) .
memb(X, [_|T]) :-
    memb(X, T) .
▲
?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
?- memb(3, [1,2,3,4]).
true .
?- memb(5, [1,2,3,4]).
false.

```

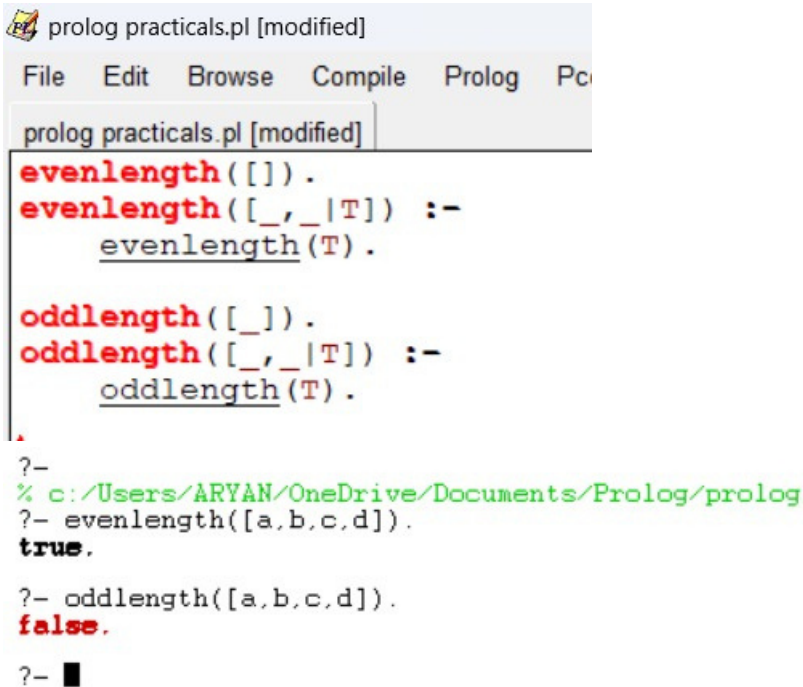
11. Write a PROLOG program to implement sumlist(L, S) so that S is the sum of a given list L.

```

prolog practicals.pl
File Edit Browse Compile Prolog Pce
prolog practicals.pl
sumlist([], 0).
sumlist([H|T], S) :-
    sumlist(T, S1),
    S is H + S1.
▲
?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/p
?- sumlist([1,2,3,4], S).
S = 10.
?-

```

12. Write a PROLOG program to implement two predicates `evenlength(List)` and `oddlength(List)` so that they are true if their argument is a list of even or odd length respectively



```
prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pc
prolog practicals.pl [modified]
evenlength( []).
evenlength([_,_|T]) :-
    evenlength(T).

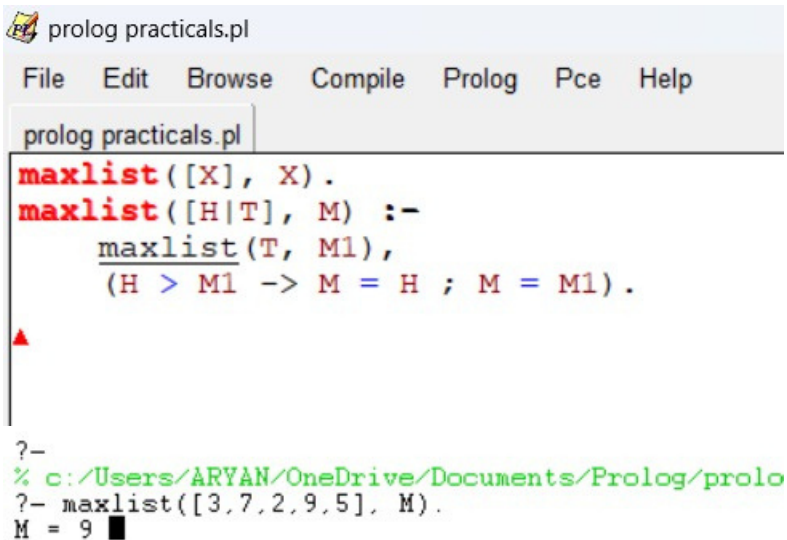
oddlength([_]).
oddlength([_,_|T]) :-
    oddlength(T).

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog
?- evenlength([a,b,c,d]).
true.

?- oddlength([a,b,c,d]).
false.

?- ■
```

13. Write a PROLOG program to implement `maxlist(L, M)` so that M is the maximum number in the list.

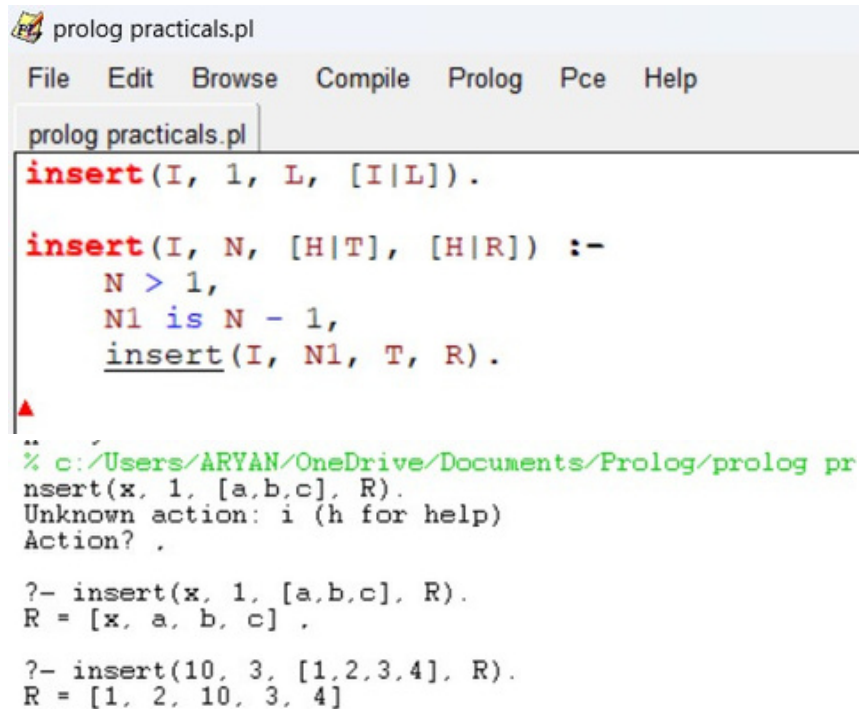


```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl
maxlist([X], X).
maxlist([H|T], M) :-
    maxlist(T, M1),
    (H > M1 -> M = H ; M = M1).

▲

?-
% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolo
?- maxlist([3,7,2,9,5], M).
M = 9 ■
```

14. Write a PROLOG program to implement `insert(I, N, L, R)` that inserts an item `I` into `N`th position of list `L` to generate a list `R`.



```
prolog practicals.pl
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl
insert(I, 1, L, [I|L]).

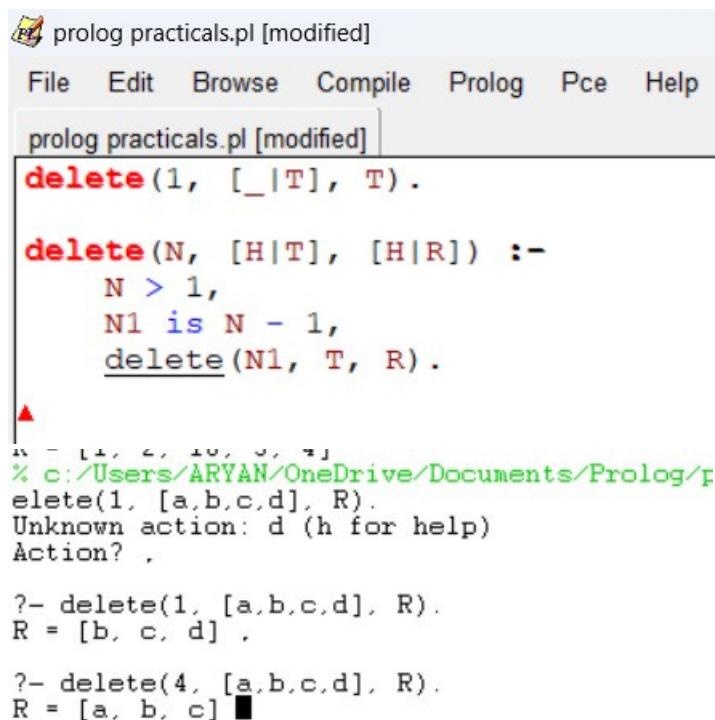
insert(I, N, [H|T], [H|R]) :-
    N > 1,
    N1 is N - 1,
    insert(I, N1, T, R).

% c:/Users/ARYAN/OneDrive/Documents/Prolog/prolog pr
nser(x, 1, [a,b,c], R).
Unknown action: i (h for help)
Action? .

?- insert(x, 1, [a,b,c], R).
R = [x, a, b, c] .

?- insert(10, 3, [1,2,3,4], R).
R = [1, 2, 10, 3, 4]
```

15. Write a PROLOG program to implement `delete(N, L, R)` that removes the element on `N`th position from a list `L` to generate a list `R`.



```
prolog practicals.pl [modified]
File Edit Browse Compile Prolog Pce Help
prolog practicals.pl [modified]
delete(1, [_|T], T).

delete(N, [H|T], [H|R]) :-
    N > 1,
    N1 is N - 1,
    delete(N1, T, R).

% c:/Users/ARYAN/OneDrive/Documents/Prolog/p
elete(1, [a,b,c,d], R).
Unknown action: d (h for help)
Action? .

?- delete(1, [a,b,c,d], R).
R = [b, c, d] .

?- delete(4, [a,b,c,d], R).
R = [a, b, c]
```