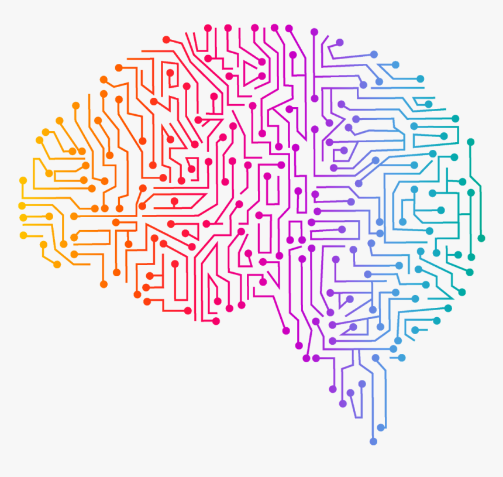
­­­­

Atma Ram Sanatan Dharma College  
University of Delhi



Artificial Intelligence

Practical File for Paper Code 32341601

Submitted By  
Sudipto Ghosh  
College Roll No. 19/78003  
BSc (Hons) Computer Science

Submitted To  
Dr Parul Jain  
Department of Computer Science

# INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| S No. | Objective | Date | Sign |
| 1 | Write a Prolog program to calculate the sum of two numbers. |  |  |
| 2 | Write a Prolog program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y. |  |  |
| 3 | Write a program in Prolog to implement factorial (N, F) where F represents the factorial of a number N. |  |  |
| 4 | Write a program in Prolog to implement generate\_fib(N,T) where T represents the Nth term of the fibonacci series. |  |  |
| 5 | Write a Prolog program to implement GCD of two numbers. |  |  |
| 6 | Write a Prolog program to implement power(Num, Pow, Ans): where Num is raised to the power Pow to get Ans. |  |  |
| 7 | Prolog program to implement multi(N1, N2, R): where N1 and N2 denotes the numbers to be multiplied and R represents the result. |  |  |
| 8 | Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not. |  |  |
| 9 | 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. |  |  |
| 10 | Write a Prolog program to implement reverse(L, R) where List L is original, and List R is the reversed list. |  |  |
| 11 | Write a program in Prolog to implement palindrome(L) which checks whether a list L is a palindrome or not. |  |  |
| 12 | Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L. |  |  |
| 13 | 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. |  |  |
| 14 | Write a Prolog program to implement nth\_element(N, L, X) where N is the desired position, L is a list and X represents the Nth element of L. |  |  |
| 15 | Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list. |  |  |
| 16 | Write a Prolog program to implement insert\_nth(I, N, L, R) that inserts an item I into Nth position of list L to generate a list R. |  |  |
| 17 | Write a Prolog program to implement delete\_nth(N, L, R) that removes the element on Nth position from a list L to generate a list R. |  |  |
| 18 | Write a program in Prolog to implement merge(L1, L2, L3) where L1 is first ordered list and L2 is second ordered list and L3 represents the merged list. |  |  |

# PRACTICAL 1

## Objective

Write a Prolog program to calculate the sum of two numbers.

## Code

sum(X, Y) :- S is X + Y,

             write(S),

             nl.

## Output

Text, letter

Description automatically generated

# PRACTICAL 2

## Objective

Write a Prolog program to implement max(X, Y, M) so that M is the maximum of 2 numbers X and Y.

## Code

max(X, Y) :- X == Y -> (format('X and Y are Equal'), nl);

             X > Y -> (Z is X, write(Z), nl);

             X < Y -> (Z is Y, write(Z), nl).

## Output

Text, letter

Description automatically generated

# PRACTICAL 3

## Objective

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

## Code

factorial(0, 1).

factorial(N, F) :- N > 0, N1 is N - 1,

                   factorial(N1, F1),

                   F is N \* F1.

## Output

Text

Description automatically generated

# PRACTICAL 4

## Objective

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

## Code

generate\_fib(0, 1).

generate\_fib(1, 1).

generate\_fib(N, T) :- N1 is N - 1,

                      generate\_fib(N1, T1),

                      N2 is N - 2,

                      generate\_fib(N2, T2),

                      T is T1 + T2.

## Output

Text

Description automatically generated

# PRACTICAL 5

## Objective

Write a Prolog program to implement GCD of two numbers.

## Code

gcd(X, Y) :- X == 0 -> write(Y);

             Y == 0 -> write(X);

             X == Y -> write(X);

             X > Y -> (Z is X - Y, gcd(Z, Y));

             Y > X -> (C is Y - X, gcd(X, C)).

## Output

Text, letter

Description automatically generated

# PRACTICAL 6

## Objective

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

## Code

power(0, Power, 0) :- Power > 0.

power(Num, 0, 1) :- Num > 0.

power(Num, Power, Ans) :- Num > 0, Power > 0,

                          P1 is Power - 1,

                          power(Num, P1, A1),

                          Ans is A1 \* Num.

## Output

Text, letter

Description automatically generated

# PRACTICAL 7

## Objective

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

## Code

multi(N, 1, N).

multi(N1, N2, R) :- T is N2 - 1,

                    multi(N1, T, Y),

                    R is Y + N1.

## Output

Text, letter

Description automatically generated

# PRACTICAL 8

## Objective

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

## Code

memb(X, [X|\_]) :- !.

memb(X, [\_|L]) :- memb(X, L).

## Output

Text

Description automatically generated

# PRACTICAL 9

## Objective

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.

## Code

conc([], L, L) :- !.

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

## Output

Text

Description automatically generated

# PRACTICAL 10

## Objective

Write a Prolog program to implement reverse(L, R) where List L is the original list, and List R is the reversed list.

## Code

conc([], L, L) :- !.

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

reverse([], []) :- !.

reverse([H|T], R) :- reverse(T, RT),

                     conc(RT, [H], R).

## Output

Text, letter

Description automatically generated

# PRACTICAL 11

## Objective

Write a program in Prolog to implement palindrome(L) which checks whether a list L is a palindrome or not.

## Code

conc([], L, L) :- !.

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

reverse([], []) :- !.

reverse([H|T], R) :- reverse(T, RT),

                     conc(RT, [H], R).

palindrome(L) :- reverse(L, L).

## Output

Text

Description automatically generated

# PRACTICAL 12

## Objective

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

## Code

sumlist([], 0).

sumlist([H|T], S) :- sumlist(T, S1),

                     S is H + S1.

## Output

Text

Description automatically generated

# PRACTICAL 13

## Objective

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.

## Code

evenlength([]) :- !.

evenlength([\_|T]) :- oddlength(T).

oddlength([\_]) :- !.

oddlength([\_|T]) :- evenlength(T).

## Output

Text, letter

Description automatically generated

# PRACTICAL 14

## Objective

Write a Prolog program to implement nth\_element(N, L, X) where N is the desired position, L is a list and X represents the Nth element of L.

## Code

nth\_element(1, [H|T], H) :- !.

nth\_element(N, [H|T], X) :- N1 is N - 1,

                            nth\_element(N1, T, X).

## Output

Text

Description automatically generated

# PRACTICAL 15

## Objective

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

## Code

max(X, Y, Z) :- X > Y, Z is X.

max(X, Y, Z) :- X < Y, Z is Y.

maxlist([], 0) :- !.

maxlist([R], R) :- !.

maxlist([H|T], R) :- maxlist(T, R1),

                     max(H, R1, R), !.

## Output

Text

Description automatically generated

# PRACTICAL 16

## Objective

Write a Prolog program to implement insert\_nth(I, N, L, R) that inserts an item I into Nth position of list L to generate a list R.

## Code

insert\_nth(E, 1, L, [E|L]) :- !.

insert\_nth(E, P, [H|L], [H|R]) :- P1 is P - 1,

                                  insert\_nth(E, P1, L, R).

## Output

Text

Description automatically generated

# PRACTICAL 17

## Objective

Write a Prolog program to implement delete\_nth(N, L, R) that removes the element on Nth position from a list L to generate a list R.

## Code

delete\_nth(1, [\_|L], L) :- !.

delete\_nth(P, [H|L], [H|R]) :- P1 is P - 1,

                               delete\_nth(P1, L, R).

## Output

Text

Description automatically generated

# PRACTICAL 18

## Objective

Write a program in Prolog to implement merge(L1, L2, L3) where L1 is first ordered list and L2 is second ordered list and L3 represents the merged list.

## Code

merge(X, [], X).

merge([], Y, Y).

merge([X|X1], [Y|Y1], [X|Z]) :- X < Y, !, merge(X1, [Y|Y1], Z).

merge([X|X1], [Y|Y1], [X,Y|Z]) :- X == Y, !, merge(X1, Y1, Z).

merge([X|X1], [Y|Y1], [Y|Z]) :- X > Y, !, merge([X|X1], Y1, Z).

## Output

Text

Description automatically generated