AI Practical File

Name : Rakesh Kumar

Roll no: 5778

sem : 6th

Sec: B

B.sc(H) Computer Science

P 1. Write a prolog program to calculate the sum of two numbers.

Ans:

```
sum(A,B):-C is A + B,write("sum is: "),write(C).
```

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

Ans:

```
max(X,Y,M):-
    (X>Y,
      M is X).
max(X,Y,M):-
    (Y>X,
      M is Y ).
max(X,Y):-
    (X>Y
    -> print(X);
   print(Y)
    ).
```

P 3. Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.

Ans:

```
factorial(N):- factorial(N, A), print(A),!.
        factorial(0,1).
        factorial(N,F):-(
            N > 0
            ->  (
                    N1 is N-1,
                    factorial(N1, F1),
                    F is N*F1
                )
            ; print("Not defined"),!
        ).
```

P 4. **Write a program in PROLOG to implement generate_fib(N,T) where T represents the Nth term of the fibonacci series.**

Ans:

```
generate_fib(N):-generate_fib(N, A), print(A),!.
generate_fib(1,1).
generate_fib(2,1).
generate_fib(N,T):-
(
   N > 0
   ->(
      N1 is N-1,
      N2 is N-2,
      generate_fib(N1, T1),
      generate_fib(N2, T2),
      T is T1 + T2
      )
   ; print("Not Defined"),!
).
```

P 5. Write a Prolog program to implement GCD of two numbers.

Ans:

```
gcd(A,B):-A1 is abs(A), B1 is abs(B), gcd(A1, B1, N1), print(N1).
gcd(0,B,N):- N is B.
gcd(A,0,N):- N is A.
gcd(A,B,N):-
   (   A = B
      ->  N is A
      ;   (   A > B
            ->  (   N1 is A-B,
                  gcd(N1,B, G),
                  N is G
               )
            ;   (   N1 is B - A,
                  gcd(A, N1, G),
                  N is G
               )
         )
   ).
```

P 6. Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.

Ans:

```
power(N, P):- power(N, P, A), print(A),!.
power(1, _, 1).
power(0, _, 0).
power(_, 0, 1).
power(N, P, A):-
(
   P > 0
   ->  (
          P1 is P - 1,
          power(N, P1, A1),
          A is N*A1
       )
   ;   (
          P1 is P + 1,
          power(N, P1, A1),
          A is 1/N*A1
       )

).
```

P 7. Prolog program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.

Ans:

```
multi(X,Y):-multi(X,Y,R), print(R).
multi(0, _, 0).
multi(_, 0, 0).
multi(X,Y,R):-
(   Y > 0
   ->  (
          Y1 is Y - 1,
          multi(X,Y1,R1),
          R is X + R1
       )
   ;   (
          Y1 is Y + 1,
          multi(X,Y1,R1),
          R is -1*X + R1
       )

).
```

P 8. Write a program in PROLOG to implement towerofhanoi (N) where N represents the number of discs

Ans:

```
toh(N):-
(
   N < 0
   ->  print("Not defined")
   ;   power(2, N, R),
       R1 is R - 1,
       write("No of steps: "),write(R1),nl,
       toh(N, "a","b","c")
).
toh(1, A,_,C):-write("Move from "),write(A),write(" to "),write(C),nl.
toh(N, A, B, C):-
(
   N1 is N-1,
   toh(N1, A,C,B),
   write("Move from "),write(A),write(" to "),write(C),nl,
   toh(N1, B,A,C)
).
```

P 9.Consider a cyclic directed graph [edge (p, q), edge (q, r), edge (q, r), edge (q, s), edge (s,t)] where edge (A,B) is a predicate indicating directed edge in a graph from a node A to a node B. Write a program to check whether there is a route from one node to another node.

Ans:

```
node(p).
node(q).
node(r).
node(s).
node(t).
edge(p,q).
edge(q,r).
edge(r,q).
edge(q,s).
edge(s,t).
path(X,Y,R):-
(
   node(X),
   node(Y),
   X \= Y
   ->  (
          edge(X,Y)
```

```
        -> R is 1
        ; (
            edge(X,Z),
            Y \= Z,
            path(Z,Y, R2),
            R2 = 1
            -> R is 1
            ; R is 0
          )
      )
   ; R is 0
).
```

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

Ans:

```
memb(A,L):-(
   memb(A,L,R),
   R = 1
   -> write(A), write(" is a member of list"),!
   ; write(A), write(" is not a member of list"),!
).
memb(H, [H|_], 1).
memb(X, [H|T], R):-
(
   X = H
   ->R is 1
   ; (
       memb(X, T, R1),
       R is R1
     )
).

/*length*/
len(L):-len(L,R),write("Length of list is: "),write(R).
len([], 0).
len([_|T], R):-
(
   len(T, R1),
   R is R1 + 1
).
/*append*/
append(X, L):-conc([X],L,R), print(R).
```

P 11.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.

Ans:

```
conc(A,B):-conc(A,B,R),write("concatenated list is: "),write(R),!.
conc([], X, X).
conc(X, [], X).
conc([H1|T1], L2, [H1|T3]):- conc(T1, L2, T3).
```

P 12.Write a Prolog program to implement reverse (L, R) where List L is original and List R is reversed list.

Ans:

```
rev(X):-rev(X,R),write("reversed list is: "),write(R).
rev(X, R):-rev(X,[],R).
rev([], X, X).
rev([H1|T1], PREV, REV):-rev(T1, [H1|PREV], REV).
/*equals*/
equals([], []):-print("yes").
equals([H1|T1],[H2|T2]):-
(
   H1 = H2
   -> equals(T1, T2)
   ; print("no")
).
```

P 13.Write a program in PROLOG to implement palindrome (L) which checks whether a list L is a palindrome or not.

Ans:

```
palindrome(A):-rev(A,R), A = R.
```

P 14.Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.

Ans:

```
lsum(L):-lsum(L,S),write("sum of list is: "),write(S).
lsum([], 0).
lsum([H|T], R):-
(
   lsum(T, R1),
   R is H + R1
).
```

P 15.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

Ans:

```
evenlength(L):-
(
   len(L, R1),
   0 is mod(R1,2)
).
oddlength(L):-
(
   len(L, R1),
   1 is mod(R1,2),
   print("true")
).
```

P 16.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.

Ans:

```
nth_element(N, L):-nth_element(N, L, X),write("element at pos
"),write(N),write(" is: "),write(X),!.
nth_element(_, [], _):-print("out of bounds"),!.
nth_element(N, [H|T], X):-
(
   N > 0
   ->  (
         N = 1
         ->X is H
         ;   (
               N1 is N - 1,
               nth_element(N1, T, X)
             )
       )
   ; print("Invalid index")
).
```

P 17.Write a program in PROLOG to implement remove_dup (L, R) where L denotes the list with some duplicates and the list R denotes the list with duplicates removed.

Ans:

```
remove_dup(L):-remove_dup(L, R),write("List after removing duplicates is:
"),write(R),!.
remove_dup([], []).
remove_dup([H|T], [H|R]):-
(
    delete_all(H, T, R1),
    remove_dup(R1, R)
).
```

P 18.Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list

Ans:

```
maxlist([H|T]):-maxlist(T, H).
maxlist([H|[]], M):-
(
    M > H
    -> print(M),!
    ; print(H),!
).
maxlist([H|T], M):-
(
    M > H
    ->maxlist(T, M)
    ; maxlist(T, H)
).
```

P 19.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.

Ans:

```
insert_nth(X, P, L):-insert_nth(X, P, L, R),print(R),!.
insert_nth(X, 1, Y, [X|Y]).
insert_nth(X, P, [H|T], [H|T1]):-
(
    P1 is P - 1,
    insert_nth(X, P1, T, T1)
).
```

P 20.Write a Program in PROLOG to implement sublist(S, L) that checks whether the list S is the sublist of list L or not. (Check for sequence or the part in the same order).

Ans:

```
sublist([],[]):-print("It is a sublist").
```

```
sublist([], [_|_]):-print("It is a sublist").
sublist([_|_], []):-print("Not a sublist").
sublist([H|T], [H1|T1]):-
(
   H = H1
   -> sublist(T, T1)
   ;  sublist([H|T], T1)
).
```

P 21.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.

Ans:

```
delete_nth(P, L):-delete_nth(P, L,R), print(R),!.
delete_nth(1, [_|T], T).
delete_nth(P, [H|T], [H|T1]):-
(
   P1 is P - 1,
   delete_nth(P1, T, T1)
).
```

P 22.Write a program in PROLOG to implement delete_all (X, L, R) where X denotes the element whose all occurrences has to be deleted from list L to obtain list R.

Ans:

```
delete_all(X, L):-delete_all(X,L,R),write("List without element "),
write(X),write(" is: "),write(R),!.
delete_all(_, [], []).
delete_all(X, [X|T], L):-delete_all(X, T, L).
delete_all(X, [H|T], [H|T1]):-delete_all(X, T, T1).
```

P 23.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.

Ans:

```
merge(X, Y):-merge(X, Y, R), write("Merged list is: "),write(R),!.
merge([], X, X).
merge(X, [], X).
merge([H1|T1], [H2|T2], [X|R]):-
(
   H1 < H2
   ->  X is H1,
      merge(T1, [H2|T2], R)
```

```
    ;   X is H2,
        merge([H1|T1], T2, R)
).
```