Name : Prateek

Roll No : 20HCS4144

Course : BSc. H Computer Science

Semester : 6th

Subject : Artificial Intelligence

Artificial Intelligence Practicals

Q1.Write a prolog program to calculate the sum of two numbers.

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

```
1 ?-
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 2 clauses
1 ?- sum(4,5,S).
S = 9.
```

Q2.Write a Prolog program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y.

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

  max(_,Y,M):- M is Y.
```

```
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 2 clauses
2 ?- max(3,4,M).
M = 4.
```

Q3.Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.

```
factorial(0,1).

 factorial(N,X):- N1 is N-1,

                  factorial(N1,X1),

                  X is X1*N.
```

```
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 1 clauses
3 ?- factorial(4,X).
X = 24
```

Q4. Write a program in PROLOG to implement generate_fib(N,T) where T represents the Nth term of the fibonacci series.

```
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.
```

```
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 4 clauses
1 ?- generate_fib(5,T).
T = 8
```

Q5.Write a Prolog program to implement GCD of two numbers.

```
gcd(M,0,M):-!.

gcd(M,N,D):-N > 0,
        X is mod(M,N),
        gcd(N,X,D).
```

```
8 ?-
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 3 clauses
8 ?- gcd(12,16,D).
D = 4
```

Q6. Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.

```
power(Num,1,Num).

power(Num,Pow,Ans):- Pow1 is Pow-1,
                     power(Num,Pow1,Ans1),
                     Ans is Ans1*Num.
```

```
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 3 clauses
1 ?- power(3,6,Ans).
Ans = 729
```

Q7.Prolog program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.

```
multi(N1,1,N1).

multi(N1,N2,Ans):-  Temp is N2-1,
                    multi(N1,Temp,Ans1),
                    Ans is Ans1+N1.
```

```
% c:/Users/Chetan/Desktop/1.pl compiled 0.00 sec, 3 clauses
1 ?- multi(5,4,Ans).
Ans = 20
```

Q8. Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not.

```
memb(X, [X|Tail]).

memb(X, [Head|Tail]):- memb(X, Tail).
```

```
1 ?- memb(3,[1,2,3,4,5,6]).
true
```

Q9. 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.

```
conc([],L,L).

conc([X|M],N,[X|Q]):- conc(M,N,Q).
```

```
% c:/Users/Chetan/Desktop/AI/q11.pl compiled 0.00 sec, 3 clause
1 ?- conc([a,b,c],[d,e,f],R).
R = [a, b, c, d, e, f].
```

Q10. Write a Prolog program to implement reverse (L, R) where List L is original and List R is reversed list.

```
reverse([H|T],R):- length(T,L),
 L>0 ->(reverse(T,R1),R is H) ;
 R is H.
```

```
| ?- reverse([a,b,c,d],R).
R = [d,c,b,a]
```

Q11. Write a program in PROLOG to implement palindrome (L) which checks whether a list L is a palindrome or not.

```
palind([]):- write('palindrome').
palind([_]):- write('palindrome').
palind(L) :- append([H|T], [H], L),palind(T) ;
 write('Not a palindrome').
```

```
yes
| ?- palind([n,i,t,i,n]).
palindrome
```

Q12.Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.

```
sumlist([],0).

sumlist([H|T],S):- sumlist(T,S1),
                   S is H+S1.
```

```
2 ?- sumlist([2,4,3,7,8],S).
S = 24.
```

Q13.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.

```
even_length([]).

even_length([_|T]):- odd_length(T).

odd_length([_]).

odd_length([_|T]):- even_length(T).
```

```
1 ?- even_length([3,8,4,1,6,9]).
true .

2 ?- odd_length([3,4,1]).
true
```

Q14.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.

```
nth_element(1,[H|T],H).

nth_element(N,[H|T],X):- N1 is N-1,
                         nth_element(N1,T,X).
```

```
1 ?- nth_element(3,[2,7,4,8,3,8,1,9],X).
X = 4
```

Q15.Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.

```
maxlist([H],H).

maxlist([H|T],M):- maxlist(T,M1),

                   H<M1 -> M is M1;

                   M is H.
```

```
1 ?- maxlist([1,6,3,9,4,7],M).
M = 9.
```

Q16.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.

```
insert(L,1,Elem,[Elem|L]):-!.

insert([],_,Elem,[Elem]).

insert([H|T],N,Elem,[H|R]):- C is N-1,

                   insert(T,C,Elem,R).
```

```
2 ?- insert([1,2,3,4,5,6],4,9,R).
R = [1, 2, 3, 9, 4, 5, 6].
```

Q17.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.

delte(1,[_|T],T).

delte(P,[X|Y],[X|R]):-

P1 is P-1, delte(P1,Y,R).

```
| ?- delte(3,[1,2,3,4,5],R).
R = [1,2,4,5] ?
```

Q18.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.

```
dmerge([],L2,L2).

dmerge(L1,[],L1).

dmerge([H1|T1],[H2|T2],[H1|T3]):- H1=<H2,
                                  dmerge(T1, [H2|T2], T3).

dmerge([H1|T1],[H2|T2],[H2|T3]):- dmerge([H1|T1], T2, T3).
```

```
|    dmerge([1,2,3,4],[6,7,8,9],T3).
T3 = [1, 2, 3, 4, 6, 7, 8, 9] .

2 ?- dmerge([2,5,1,7,3],[6,7,8,9],T3).
T3 = [2, 5, 1, 6, 7, 3, 7, 8, 9]
```