KALINDI COLLEGE

UNIVERSITY OF DELHI

NAME: PRACHI AGGARWAL

COURSE: BSC HONS COMPUTER SCIENCE

COLLEGE ROLL NUMBER: 21570015

UNIVERSITY ROLL NUMBER: 21033570042

SUBJECT: ARTIFICIAL INTELLIGENCE

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

   Code:

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

   Output :

   ```
   ?- sum(3,6,X).
   X = 9.

   ?- sum(21,6,27).
   true.

   ?- sum(12,8,21).
   false.
   ```

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

   Code:

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

   Output :

   ```
   ?- max(12,4,X).
   X = 12 ;
   false.

   ?- max(2,55,X).
   X = 55 ;
   false.
   ```

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

   Code:

   ```
   fact(0,1).
   fact(1,1).
   fact(N,X):-N>0,N1 is N-1,fact(N1,X2),X is X2*N.
   ```

   Output :

   ```
   ?- fact(3,X).
   X = 6 ,

   ?- fact(5,X).
   X = 120 ,

   ?- fact(4,24).
   true .
   ```

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

Code:

```
fibb(0,0).
fibb(1,1).
fibb(N,X):-N>1,N1 is N-1, N2 is N-2, fibb(N1,X1),fibb(N2,X2), X is X1 + X2.
```

Output :

```
?- fibb(6,X).
X = 8 .

?- fibb(8,X).
X = 21 .
```

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

Code:

```
gcd(X,X,X).
gcd(X,Y,D):- X<Y,Y1 is Y-X, gcd(X,Y1,D).
gcd(X,Y,D):- X>Y,gcd(Y,X,D)
```

Output :

```
?- gcd(20,100,X).
X = 20 .

?- gcd(81,90,X).
X = 9 .

?-
```

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

Code:

```
power(_,0,1).
power(N,P,X):- P>0,P2 is P-1,power(N,P2,X1),X is X1*N.
```

Output :

Output :

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

?- power(4,3,X).
X = 64 .

?- power(3,3,27).
true
```

7. **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(N1,N2,X):- X is N1*N2.
```

Output :

```
?- multi(5,8,X).
X = 40.

?- multi(2,12,X).
X = 24.

?- multi(3,5,15).
true.
```

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

Code:

```
mem(X,[X|_]).
mem(X,[_|T]):-mem(X,T).
```

Output :

```
?- mem(a,[b,d,a,r,w]).
true ;
false.
```

```
?- mem(1,[4,5,2,6]).
false.
```

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.**

Code:

```
concatenate([],L2,L2).
concatenate([H|T],L2,[H|L3]):-concatenate(T,L2,L3).
```

Output :

```
?- concatenate([1,2,3],[4,5,6],X).
X = [1, 2, 3, 4, 5, 6].

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

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

Code:

```
rev([],[]).
rev([H|T],R):-rev(T,R1),concatenate(R1,[H],R).
```

Output :

```
?- rev([1,2,3,4],X).
X = [4, 3, 2, 1].

?- rev([a,b,c],[c,b,a]).
true.
```

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

Code:

```
pal(L):-rev(L,L).
```

Output :

```
?- pal([1,2,2,1]).
true.

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

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

Code:

```
sum([],0).
sum([H|T],S):-sum(T,S1),S is S1+H.
```

Output :

```
?- sum([2,3,4],X).
X = 9.

?- sum([2,3,4],9).
true.

?- sum([2,3,4],4).
false.
```

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.

Code:

```
odd([_]).
odd([_,_|T]):-odd(T).

even([]).
even([_,_|T]):-even(T).
```

Output :

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

?- odd([1,2,3]).
true .

?- odd([1,2,3,4]).
false.

?- even([1,2,3]).
false.
```

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.

Code:

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

Output :

```
?- nth_element(3,[1,2,4,5],X).
X = 4 .

?- nth_element(3,[1,2,4,5],X).
X = 4 .

?- nth_element(5,[a,b,c,d,e,f,g],e).
true .
```

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

Code:

```
max(X,Y,M):-X>Y,M is X.
max(X,Y,M):-X<Y,M is Y.
max(X,X,M):-M is X.
maxlist([X],X).
maxlist([X,Y|T],M):-
   maxlist([Y|T],MT),
   max(X,MT,M).
```

Output :

```
?- maxlist([3,2,5,13],X).
X = 13 .

?- maxlist([3,2,5,13],13).
true .
```

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.

Code:

```
insert(1,N,L,[N|L]).
insert(I,N,[H|T],[H|X]):-I1 is I-1, insert(I1,N,T,X).
```

Output :

```
?- maxlist([3,2,5,13],X).
X = 13 .

?- maxlist([3,2,5,13],13).
true .

?-  insert(3,5,[a,b,c,d,e],X).
X = [a, b, 5, c, d, e] .

?- insert(1,12,[1,2,3,4],X).
X = [12, 1, 2, 3, 4] .
```

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.

Code:

```
delete_nth(1,[_|Xs],Xs).
delete_nth(N,[X|Xs],[X|R]):-N>1,N1 is N-1,delete_nth(N1,Xs,R).
```

Output :

```
?- delete_nth(3,[a,b,c,d,e],L).
L = [a, b, d, e] .

?- delete_nth(1,[a,b,c,d,e],L).
L = [b, c, d, e] .

?- delete_nth(5,[a,b,c],L).
false.
```

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.

Code:

```
merge_list([],L,L).
merge_list(L,[],L).
merge_list([X|R1],[Y|R2],[X|R3]):-X<Y,!,merge_list(R1,[Y|R2],R3).
merge_list(L1,[Y|R2],[Y|R3]):-merge_list(L1,R2,R3).
```

Output :

```
?- merge_list([1,3,5,6],[2,4,7],L).
L = [1, 2, 3, 4, 5, 6, 7] .

?- merge_list([32,56,78,88],[12,43,56,76],L).
L = [12, 32, 43, 56, 56, 76, 78, 88] .
```