Artificial Intelligence

PRACTICAL FILE

SUBMITTED BY:

K A SHANKAR NARAYAN

19020570018

Q1. Sum of 2 number using prolog.

sum(X,Y,Z):- Z is X+Y.

OUTPUT:

```
?- sum(4,6,X).
X = 10.
```

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(X,Y,M):-Y>=X,M is Y.

OUTOUT:

```
?- max(7,3,X).
X = 7
```

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

fact(0,1).

fact(N,X):-N1 is N-1,fact(N1,Y),X is Y*N,!.

OUTPUT:

```
?- fact(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.

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

OUTPUT:

```
?- fib(6,X).
X = 5.
```

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

```
gcd(0,A,A):-!.
gcd(A,0,A):-!.
gcd(A,B,R):-B1 is mod(A,B),gcd(B,B1,R).
```

OUTPUT:

```
?- gcd(4,8,X).
X = 4.
```

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

power(X,0):- !.

power(Num,Pow,Ans):-Ans is Num^Pow.

OUTPUT:

```
?- pow(3,4,X).
X = 81.
```

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(X,0).

multi(N1,N2,R):- R is N1*N2.

OUTPUT:

```
?- multi(3,4,X).
X = 12.
```

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

```
memb(X,[X|T]).
memb(X,[H|T]):-memb(X,T).
```

OUTPUT:

```
?- memb(3,[1,2,3,4]).
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|L1],L2,[X|L3]):-conc(L1,L2,L3).
```

OUTPUT:

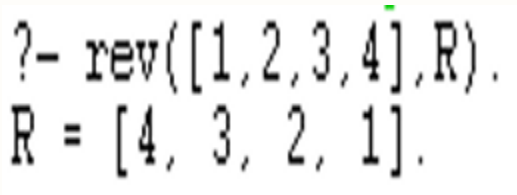
```
?- conc([1,2],[3,4,5],I).
I = [1, 2, 3, 4, 5].
```

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

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

```
app([X|L1],L2,[X|L3]):- app(L1,L2,L3).
rev([],[]).
rev([H|T],R):- rev(T,L1),app(L1,[H],R).
```

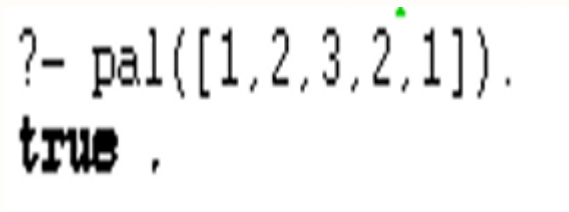
OUTPUT:



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

```
app([],L,L).
app([X|L1],L2,[X|L3]):- app(L1,L2,L3).
pal([]).
pal([_]).
pal(P):-app([H|T],[H],P),pal(T).
```

OUTPUT:



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.
```

OUTPUT:

```
?- sumlist([1,2,3],S).
S = 6.
```

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.

```
evelength([]).
evelength([_|[_|List]]):-evelength(List).
oddlength([_]).
oddlength([_|[_|List]]):-oddlength(List).
```

OUTPUT:

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

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

OUTPUT:

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

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

```
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:

```
?- maxlist([1,2,5,4,3],M).
M = 5.
```

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.

```
insertn(Item,List,1,[Item|List]).

insertn(Item,[H|List],Pos,[H|Result]):-Pos1 is Pos-1,insertn(Item,List,Pos1,Result).
```

OUTPUT:

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

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

```
removen([_|List],1,List).
removen([H|List],Pos,[H|Result]):-Pos1 is Pos-1, removen(List,Pos1,Result).
```

OUTPUT:

```
?- removen([1,2,3,4],3,L).
L = [1, 2, 4]
```

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.

```
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:

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