ARTIFICIAL INTELLIGENCE

PRACTICAL FILE

TANYA CHAWLA

19/78006

EXAMINATION ROLL No.: 19003570004

B.Sc.(Hons.)COMPUTER SCIENCE

SEMESTER: VI

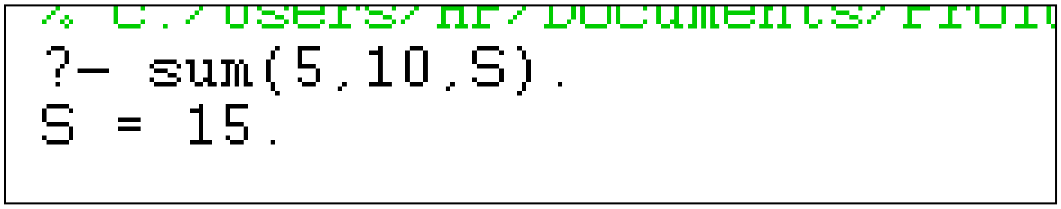
1. Write a prolog program to calculate the sum of two numbers.

A1.

CODE:

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

OUTPUT:



```
% c:/Users/hp/Documents/Prol
?- sum(5,10,S).
S = 15.
```

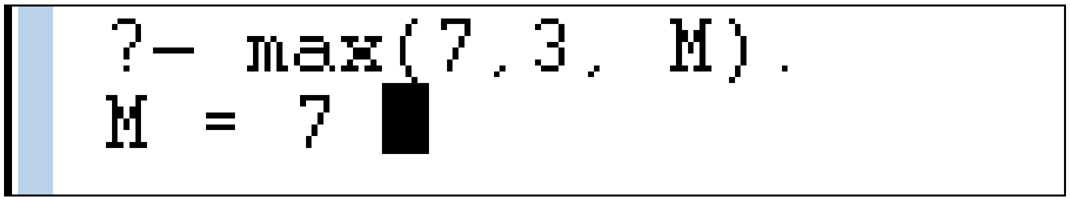
Q2. Write a Prolog program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y.

A2.

CODE:

```
max(X,Y,M):-X>Y,M is X.
max(X,Y,M):-Y>X,M is Y.
max(X,Y,M):- X=Y, write("they are equal").
```

OUTPUT:



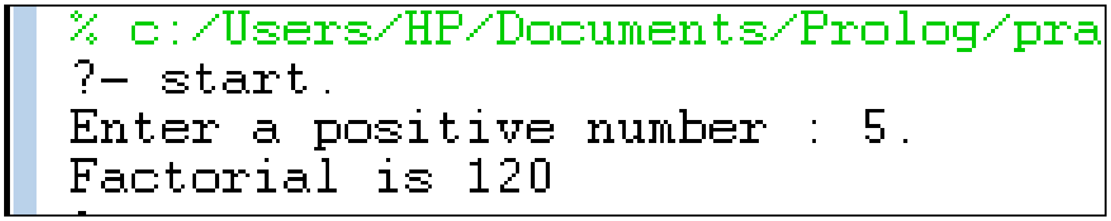
```
?- max(7,3, M).
M = 7
```

3. Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.
A3.

CODE:

```
start:-write('Enter a positive number : '),read(N1),F is 1,fac(N1,F).
fac(0,F):-write('Factorial is '),write(F).
fac(N,F):-N\=0,
        NewF is F*N,
        NewN is N-1,
        fac(NewN,NewF).
fac(1,F):-write('Factorial is ',F).
```

OUTPUT:



```
% c:/Users/HP/Documents/Prolog/pra
?- start.
Enter a positive number : 5.
Factorial is 120
```

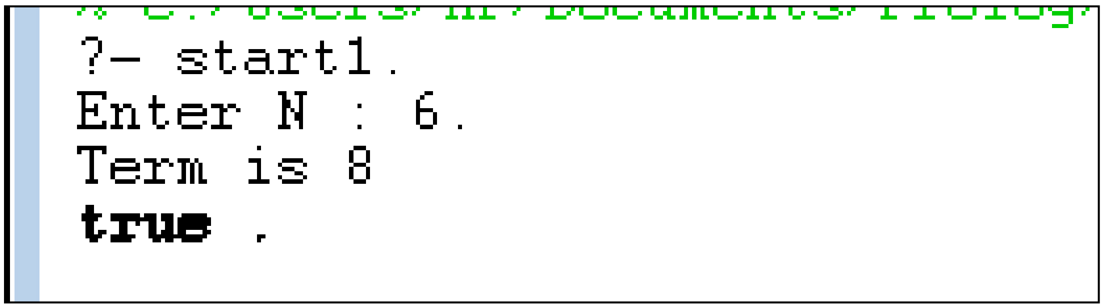
4. Write a program in PROLOG to implement generate_fib(N,T) where T represents the Nth term of the fibonacci series.

A4.

CODE:

```
start1:-write('Enter N : '),read(N), fibo(N,T), write('Term is '),write(T).
fibo(0,0).
fibo(1,1).
fibo(N,T):-N>1,
        N1 is N-1,
        fibo(N1,R1),
        N2 is N-2,
        fibo(N2,R2),
        T is R1+R2.
```

OUTPUT:



```
?- start1.
Enter N : 6.
Term is 8
true .
```

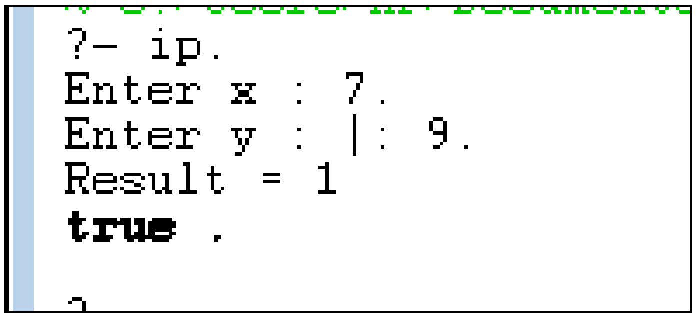
5. Write a Prolog program to implement GCD of two numbers.

A5.

CODE:

```
ip:-
        write("Enter x : "),
        read(X),
        write("Enter y : "),
        read(Y),
        gcd(X,Y).

gcd(X,X):-
        write("Result = "), write(X).

gcd(0,_):-
        write("Result = 0").

gcd(_,0):-
        write("Result = 0").

gcd(X,Y):-
        X>=Y, Xn is X-Y, gcd(Xn,Y); Xn is Y-X, gcd(X,Xn).
```

OUTPUT:



```
?- ip.
Enter x : 7.
Enter y : |: 9.
Result = 1
true .
```

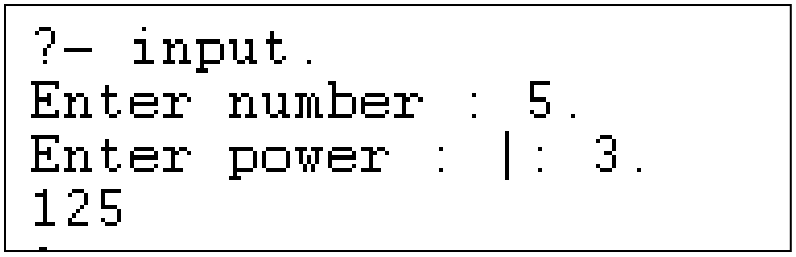
6. Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.

A6.

CODE:

```
input:-
        write("Enter number : "),
        read(Num),
        write("Enter power : "),
        read(Pow),
        Ans is 1,
        pwr(Num,Pow,Ans).

pwr(_,0,Ans):-write(Ans).

pwr(Num,Pow,Ans):-
        Pow2 is Pow-1, NewAns is Ans*Num, pwr(Num,Pow2,NewAns).
```

OUTPUT:



```
?- input.
Enter number : 5.
Enter power : |: 3.
125
```

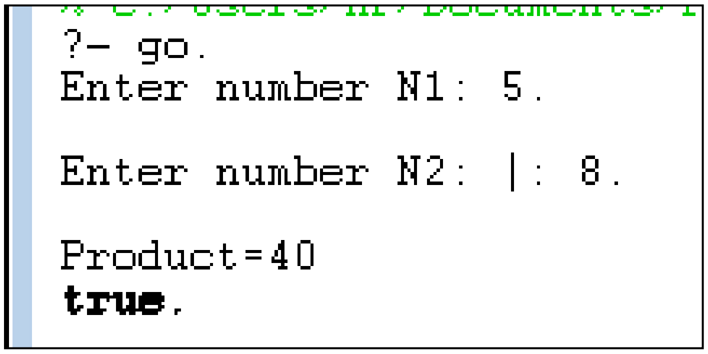
7. Prolog program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.

A7.

CODE:

```
go:-
        write("Enter number N1: "),read(N1),nl,
        write("Enter number N2: "),read(N2),nl,
        multi(N1,N2).

multi(N1,N2):-
        R is N1 * N2,
        write("Product="),write(R).
```

OUTPUT:



```
?- go.
Enter number N1: 5.

Enter number N2: |: 8.

Product=40
true.
```

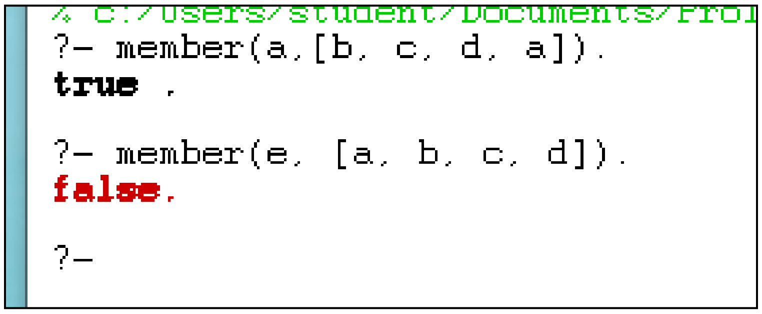
8. Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not.

A8.

CODE:

```
member(X,[X|_]).
member(X,[Y|L]):- member(X,L).
```

OUTPUT:



```
% c:/Users/student/Documents/Prol
?- member(a,[b, c, d, a]).
true .

?- member(e, [a, b, c, d]).
false.

?-
```

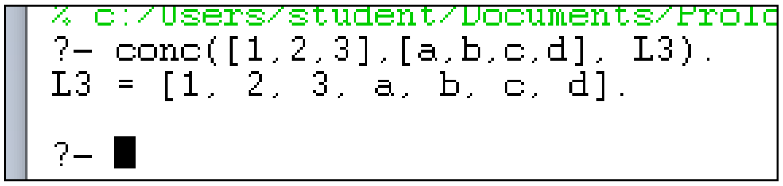
9. Write a Prolog program to implement conc (L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.

A9.

CODE:

```
conc([], List, List).
conc([X|L1],L2,[X|L3]):- conc(L1, L2, L3).
```

OUTPUT:



```
% c:/Users/student/Documents/Prolo
?- conc([1,2,3],[a,b,c,d], L3).
L3 = [1, 2, 3, a, b, c, d].

?- 
```

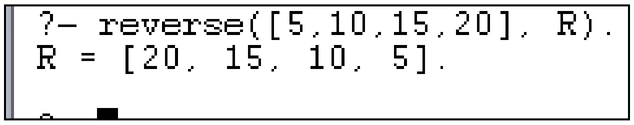
10. Write a Prolog program to implement reverse (L, R) where List L is original and List R is reversed list.

A10.

CODE:

```
append([],L,L).
append([X|L1],L2,[X|L3]):- append(L1,L2,L3).
reverse([],[]).
reverse([H|T],R):-reverse(T,L1),append(L1,[H],R).
```

OUTPUT:



```
?- reverse([5,10,15,20], R).
R = [20, 15, 10, 5].
```

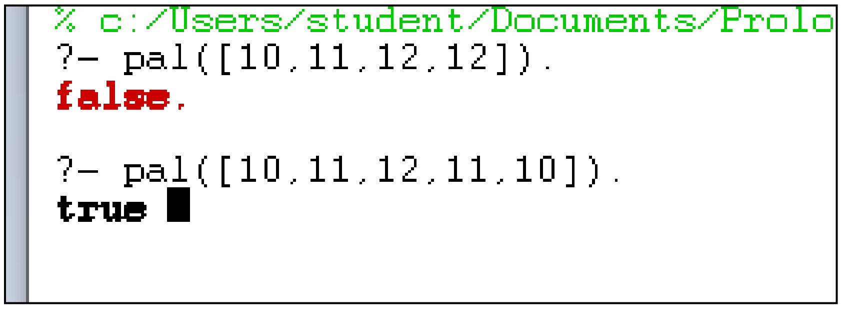
11. Write a program in PROLOG to implement palindrome (L) which checks whether a list L is a palindrome or not.

A11.

CODE:

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

OUTPUT:



```
% c:/Users/student/Documents/Prolo
?- pal([10,11,12,12]).
false.

?- pal([10,11,12,11,10]).
true
```

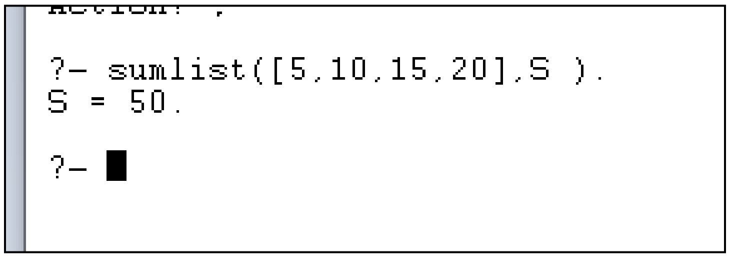
12. Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.

A12.

CODE:

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

OUTPUT:



```
?- sumlist([5,10,15,20],S ).
S = 50.

?- 
```

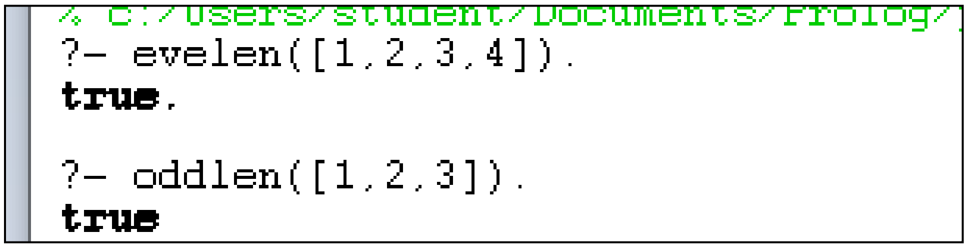
13. Write a Prolog program to implement two predicates evenlength(List) and oddlength(List) so that they are true if their argument is a list of even or odd length respectively.

A13.

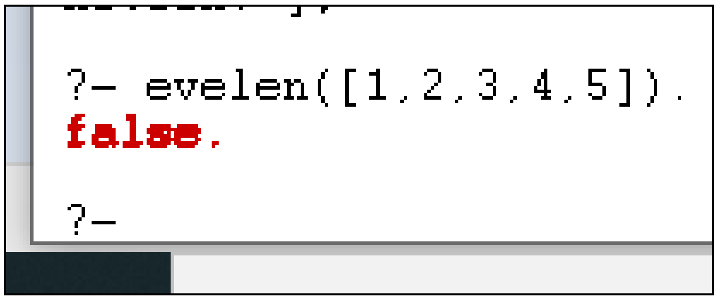
CODE:

```
evelen([]).
evelen([_|[_|List]]):- evelen(List).
oddlen([_]).
oddlen([_|[_|List]]):- oddlen(List).
```

OUTPUT:



```
% c:/Users/student/Documents/Prolog/
?- evelen([1,2,3,4]).
true.

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



```
?- evelen([1,2,3,4,5]).
false.

?-
```

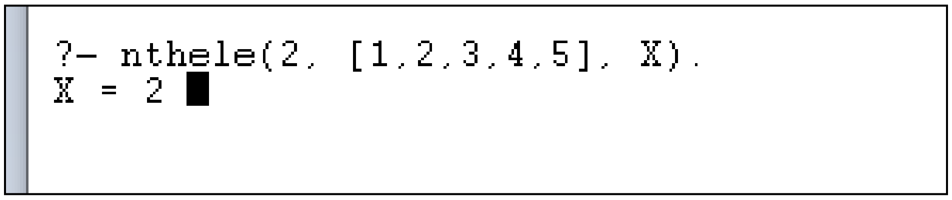
14. Write a Prolog program to implement nth_element (N, L, X) where N is the desired position, L is a list and X represents the Nth element of L.

A14.

CODE:

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

OUTPUT:



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

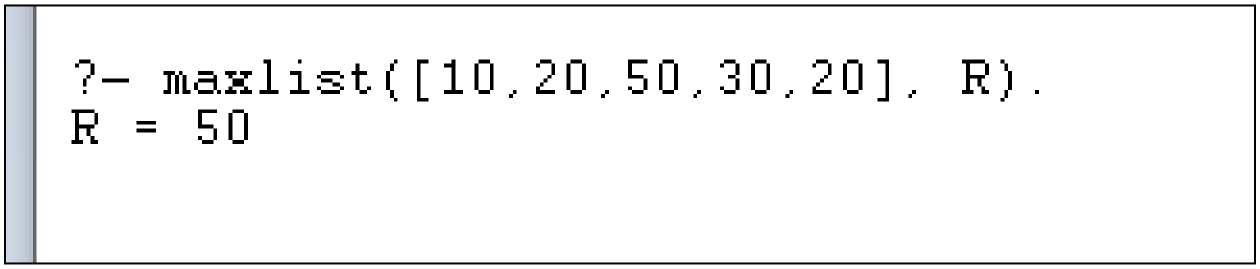
15. Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.

A15.

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:



```
?- maxlist([10,20,50,30,20], R).
R = 50
```

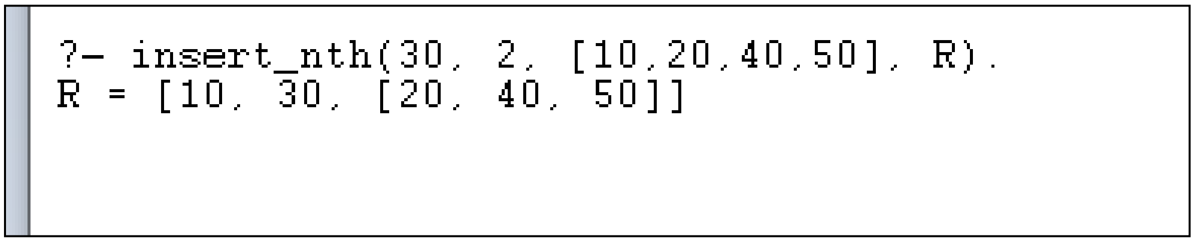
16. Write a prolog program to implement insert_nth (I, N, L, R) that inserts an item I into Nth position of list L to generate a list R.

A16

CODE:

```
insert_nth(I,1,List,[I,List]).
insert_nth(I,N,[H|T],[H|R]):- N1 is N-1, insert_nth(I,N1,T,R).
```

OUTPUT:



```
?- insert_nth(30, 2, [10,20,40,50], R).
R = [10, 30, [20, 40, 50]]
```

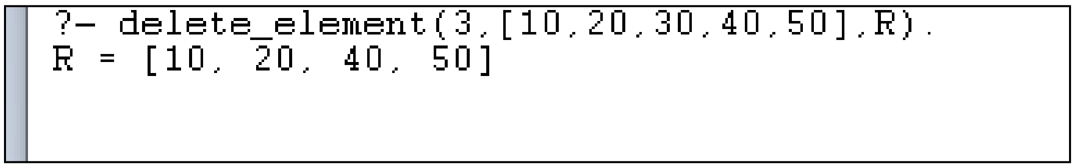
17. Write a Prolog program to implement delete_nth (N, L, R) that removes the element on Nth position from a list L to generate a list R.

A17.

CODE:

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

OUTPUT:



```
?- delete_element(3,[10,20,30,40,50],R).
R = [10, 20, 40, 50]
```

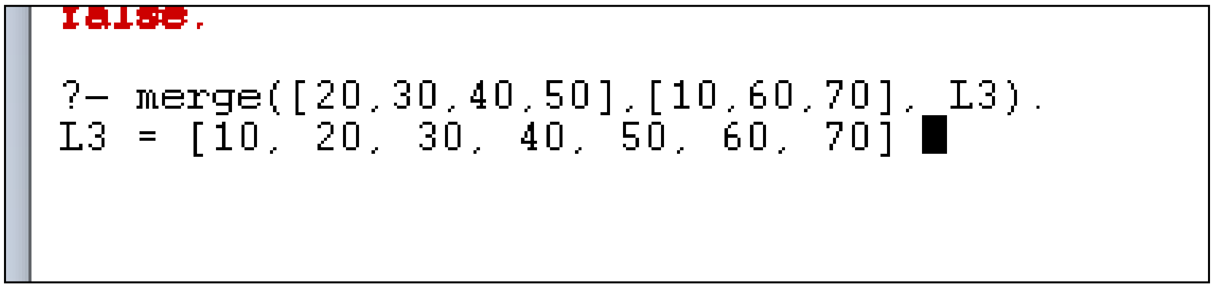
18. Write a program in PROLOG to implement merge (L1, L2, L3) where L1 is first ordered list and L2 is second ordered list and L3 represents the merged list.

A18.

CODE:

```
merge([],[],[]).
merge([],L2,L2).
merge(L1,[],L1).
merge([H1|T1],[H2|T2],[H1|T3]):- H1=<H2, merge(T1,[H2|T2],T3).
merge([H1|T1],[H2|T2],[H2|T3]):- merge([H1|T1],T2,T3).
```

OUTPUT:



```
false.

?- merge([20,30,40,50],[10,60,70], L3).
L3 = [10, 20, 30, 40, 50, 60, 70]
```