AI PRACTICAL FILE

Abhineet Raman(2002078)

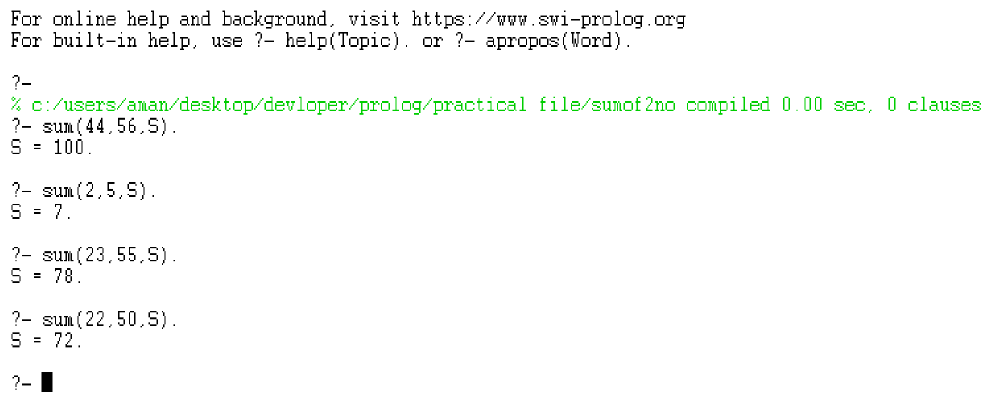
B.SC.(H.) COMPUTER SCIENCE Semester: 6

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

Knowledge Base:

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

Output:



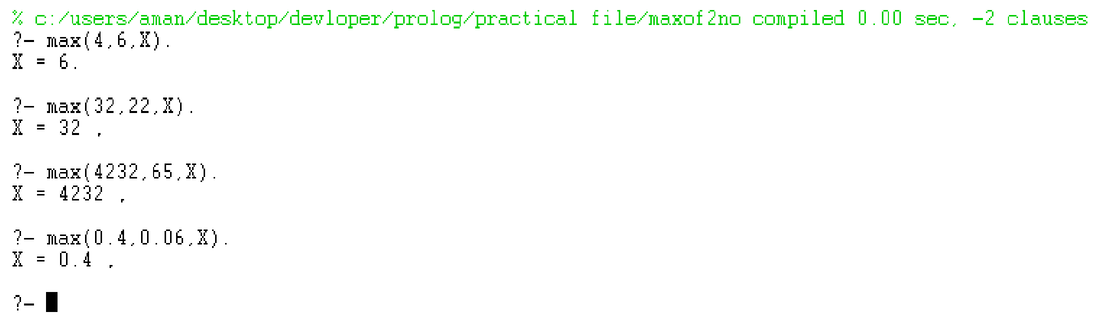
```
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% c:/users/aman/desktop/devloper/prolog/practical file/sumof2no compiled 0.00 sec, 0 clauses
?- sum(44,56,S).
S = 100.

?- sum(2,5,S).
S = 7.

?- sum(23,55,S).
S = 78.

?- sum(22,50,S).
S = 72.

?- █
```

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

Knowledge Base:

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

Output:



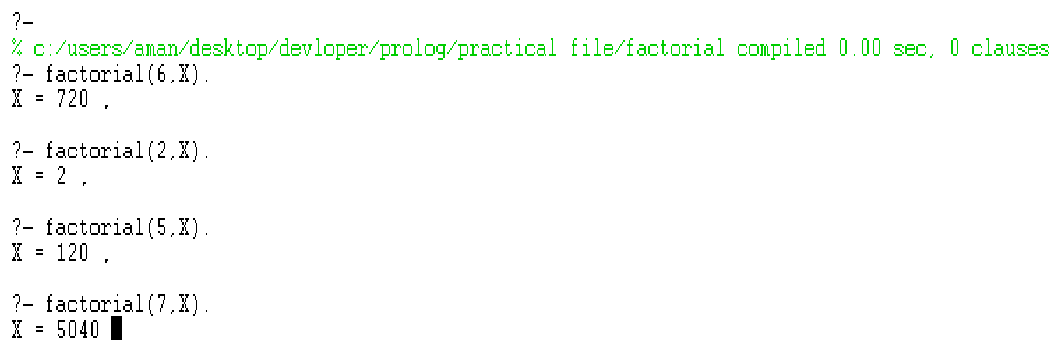
```
% c:/users/aman/desktop/devloper/prolog/practical file/maxof2no compiled 0.00 sec, -2 clauses
?- max(4,6,X).
X = 6.

?- max(32,22,X).
X = 32 ,

?- max(4232,65,X).
X = 4232 ,

?- max(0.4,0.06,X).
X = 0.4 ,

?- █
```

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

Knowledge Base:

factorial(0,1). factorial(N,X):- N1 is N-1, factorial(N1,X1), X is X1*N

Output:



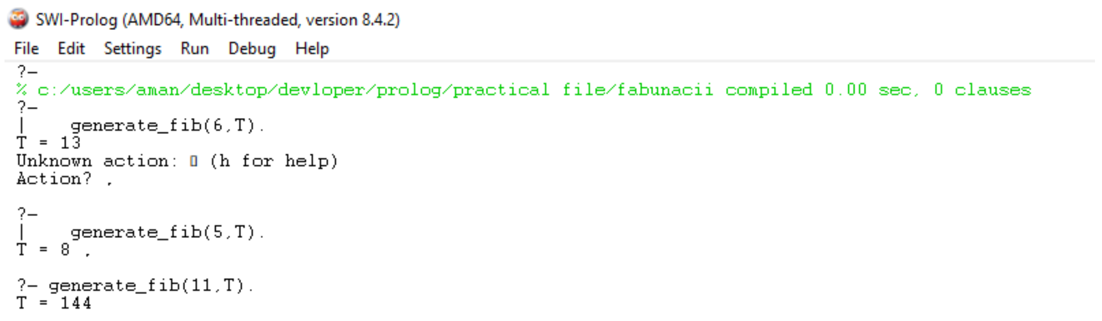
```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/factorial compiled 0.00 sec, 0 clauses
?- factorial(6,X).
X = 720 ,

?- factorial(2,X).
X = 2 ,

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

?- factorial(7,X).
X = 5040
```

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

Knowledge Base:

```
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.
```

Output:



SWI-Prolog (AMD64, Multi-threaded, version 8.4.2)
File Edit Settings Run Debug Help

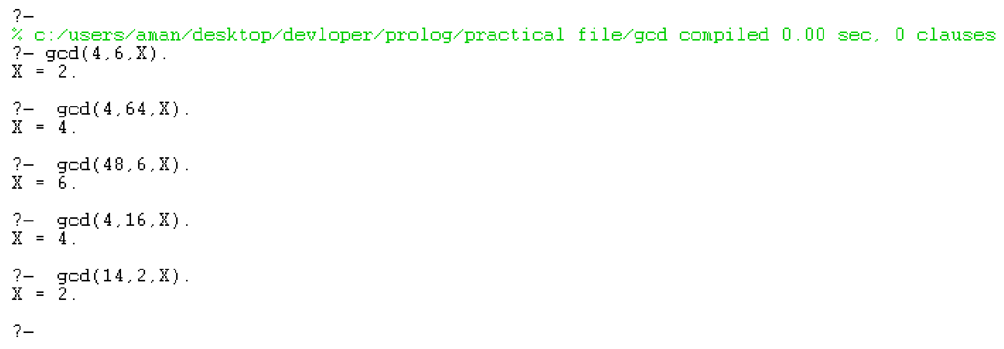
```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/fabunacii compiled 0.00 sec, 0 clauses
?-
|    generate_fib(6,T).
T = 13
Unknown action: □ (h for help)
Action? ,

?-
|    generate_fib(5,T).
T = 8 ,

?- generate_fib(11,T).
T = 144
```

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

Knowledge Base:

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

Output:



```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/gcd compiled 0.00 sec, 0 clauses
?- gcd(4,6,X).
X = 2.

?-  gcd(4,64,X).
X = 4.

?-  gcd(48,6,X).
X = 6.

?-  gcd(4,16,X).
X = 4.

?-  gcd(14,2,X).
X = 2.

?-
```

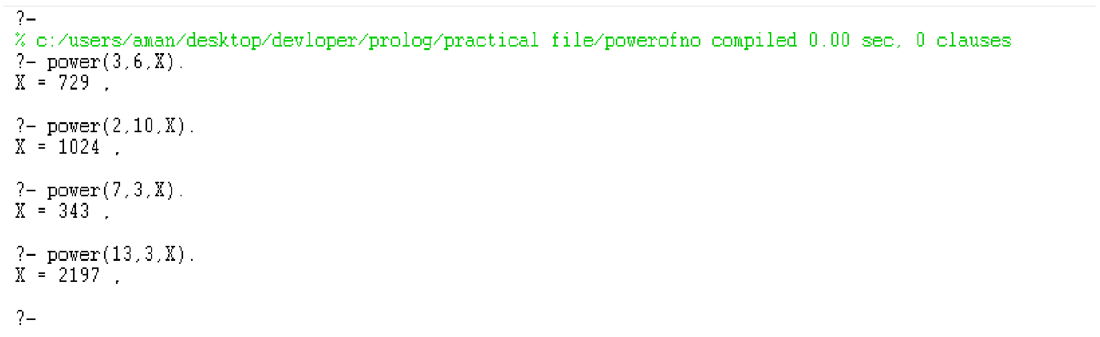
6. Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.

Knowledge Base:

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

Ans is Ans1*Num.

Output:



```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/powerofno compiled 0.00 sec, 0 clauses
?- power(3,6,X).
X = 729 ,

?- power(2,10,X).
X = 1024 ,

?- power(7,3,X).
X = 343 ,

?- power(13,3,X).
X = 2197 ,

?-
```

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

Knowledge Base:

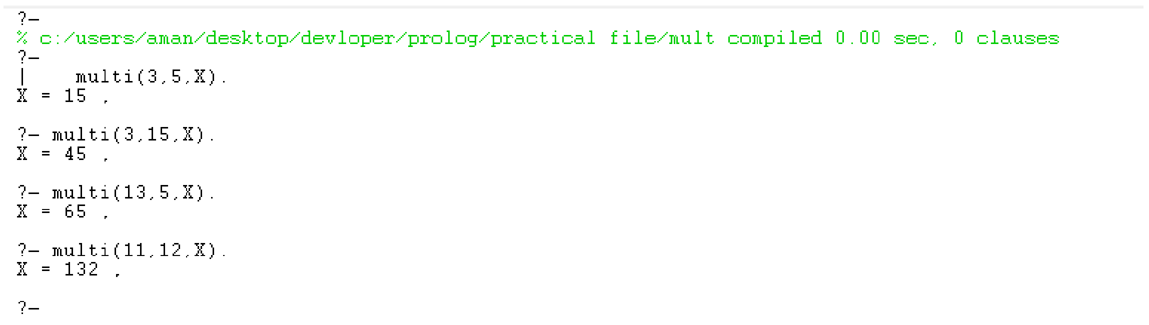
multi(N1,1,N1).

multi(N1,N2,Ans):- Temp is N2-1,

multi(N1,Temp,Ans1),

Ans is Ans1+N1

Output:



```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/mult compiled 0.00 sec, 0 clauses
?-
|    multi(3,5,X).
X = 15 ,

?- multi(3,15,X).
X = 45 ,

?- multi(13,5,X).
X = 65 ,

?- multi(11,12,X).
X = 132 ,

?-
```

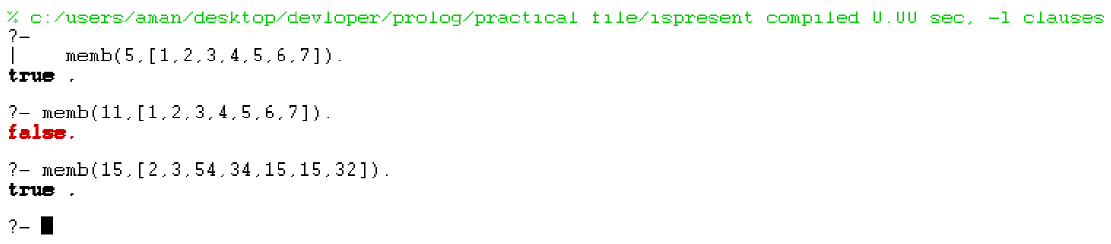
8. Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not.

Knowledge Base:

memb(X, [X|Tail]).

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

Output:



```
% c:/users/aman/desktop/devloper/prolog/practical file/ispresent compiled 0.00 sec, -1 clauses
?-
|    memb(5,[1,2,3,4,5,6,7]).
true .

?- memb(11,[1,2,3,4,5,6,7]).
false.

?- memb(15,[2,3,54,34,15,15,32]).
true .

?- 
```

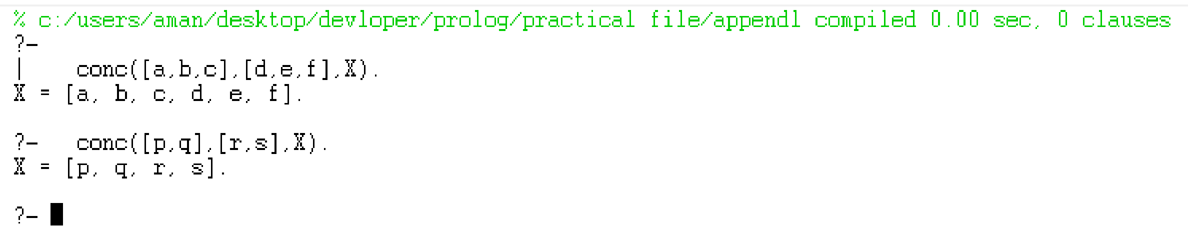
9. Write a Prolog program to implement conc (L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3

Knowledge Base:

conc([],L,L).

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

Output:



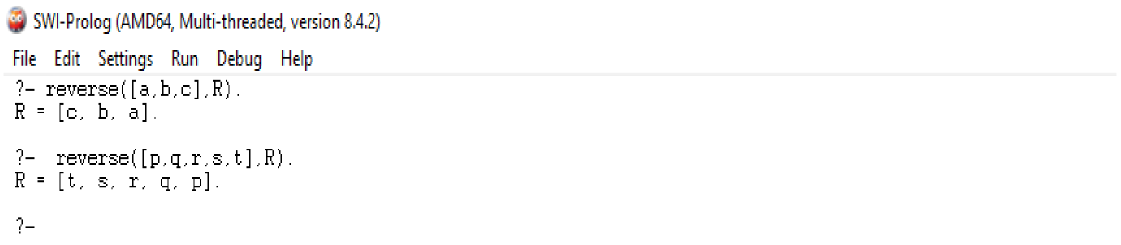
```
% c:/users/aman/desktop/devloper/prolog/practical file/append1 compiled 0.00 sec, 0 clauses
?-
|    conc([a,b,c],[d,e,f],X).
X = [a, b, c, d, e, f].

?-   conc([p,q],[r,s],X).
X = [p, q, r, s].

?- 
```

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

Knowledge Base:

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

Output:


SWI-Prolog (AMD64, Multi-threaded, version 8.4.2)
File Edit Settings Run Debug Help

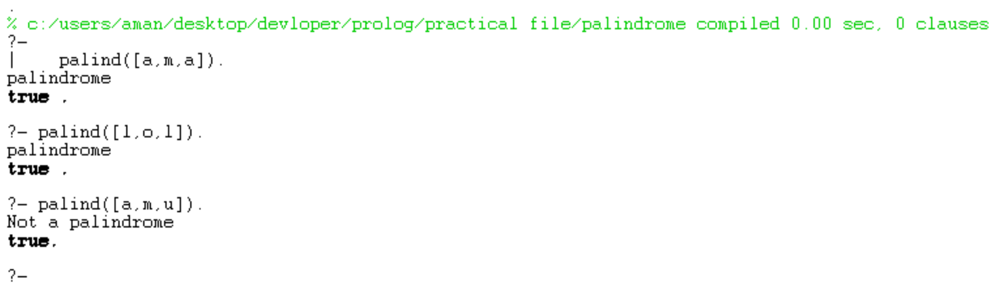
```
?- reverse([a,b,c],R).
R = [c, b, a].

?-  reverse([p,q,r,s,t],R).
R = [t, s, r, q, p].

?-
```

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

Knowledge Base:

```
palind([]):- write('palindrome').

palind([_]):- write('palindrome').

palind(L) :- append([H|T], [H], L),

   palind(T) ; write('Not a palindrome').
```

Output:



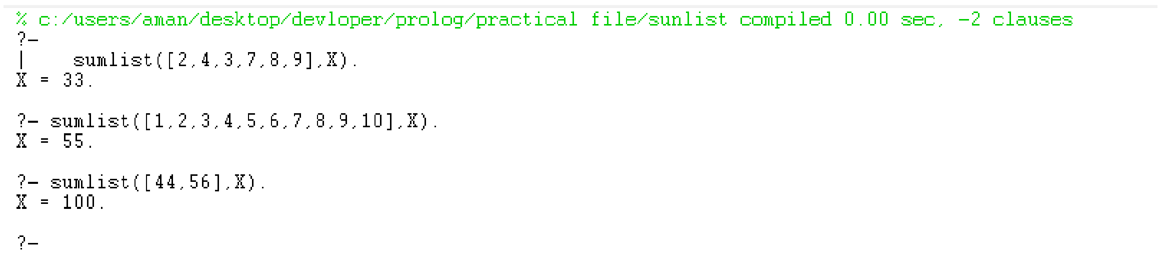
```
% c:/users/aman/desktop/devloper/prolog/practical file/palindrome compiled 0.00 sec, 0 clauses
?-
|    palind([a,m,a]).
palindrome
true .

?- palind([l,o,l]).
palindrome
true .

?- palind([a,m,u]).
Not a palindrome
true.

?-
```

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

Knowledge Base:

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

Output:



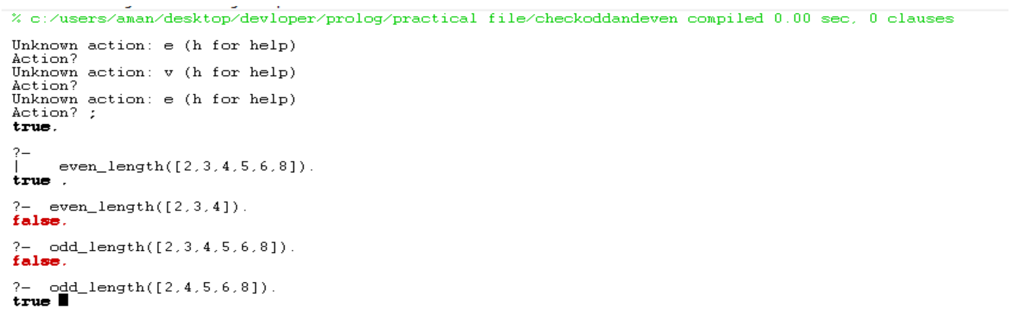
```
% c:/users/aman/desktop/devloper/prolog/practical file/sunlist compiled 0.00 sec, -2 clauses
?-
|    sumlist([2,4,3,7,8,9],X).
X = 33.

?- sumlist([1,2,3,4,5,6,7,8,9,10],X).
X = 55.

?- sumlist([44,56],X).
X = 100.

?-
```

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

Knowledge Base:

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

Output:


```
% c:/users/aman/desktop/devloper/prolog/practical file/checkoddandeven compiled 0.00 sec, 0 clauses

Unknown action: e (h for help)
Action?
Unknown action: v (h for help)
Action?
Unknown action: e (h for help)
Action? ;
true.

?-
|    even_length([2,3,4,5,6,8]).
true .

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

?-  odd_length([2,3,4,5,6,8]).
false.

?-  odd_length([2,4,5,6,8]).
true
```

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.

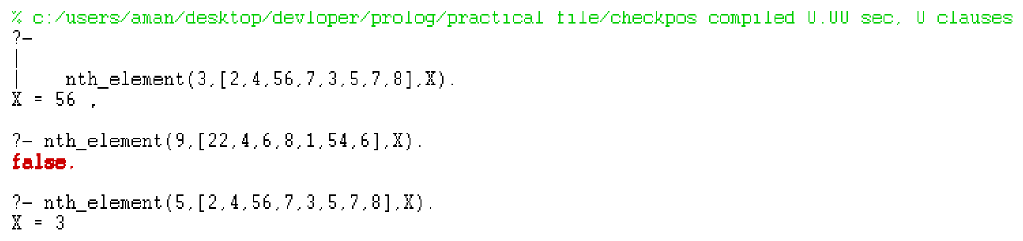
Knowledge Base:

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

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

nth_element(N1,T,X).

Output:


```
% c:/users/aman/desktop/devloper/prolog/practical file/checkpos compiled 0.00 sec, 0 clauses
?-
|
|    nth_element(3,[2,4,56,7,3,5,7,8],X).
X = 56 .

?- nth_element(9,[22,4,6,8,1,54,6],X).
false.

?- nth_element(5,[2,4,56,7,3,5,7,8],X).
X = 3
```

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

Knowledge Base:

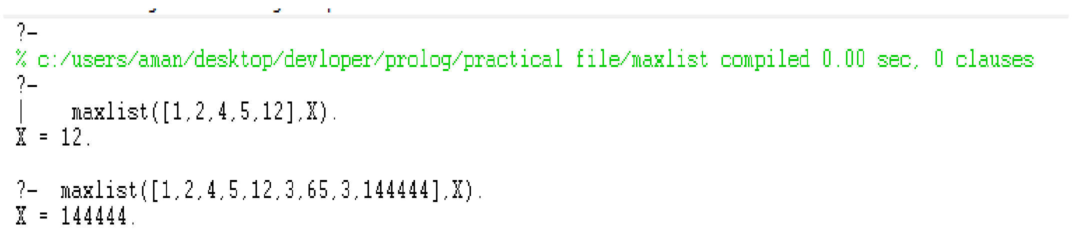
maxlist([H],H).

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

H M is M1;

M is H.

Output:



```
?-
% c:/users/aman/desktop/devloper/prolog/practical file/maxlist compiled 0.00 sec, 0 clauses
?-
|    maxlist([1,2,4,5,12],X).
X = 12.

?-  maxlist([1,2,4,5,12,3,65,3,144444],X).
X = 144444.
```

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.

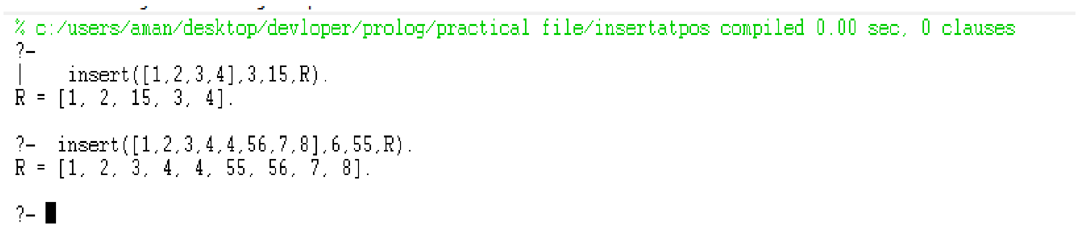
Knowledge Base:

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).

Output:



```
% c:/users/aman/desktop/devloper/prolog/practical file/insertatpos compiled 0.00 sec, 0 clauses
?-
|    insert([1,2,3,4],3,15,R).
R = [1, 2, 15, 3, 4].

?-  insert([1,2,3,4,4,56,7,8],6,55,R).
R = [1, 2, 3, 4, 4, 55, 56, 7, 8].

?-
```

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.

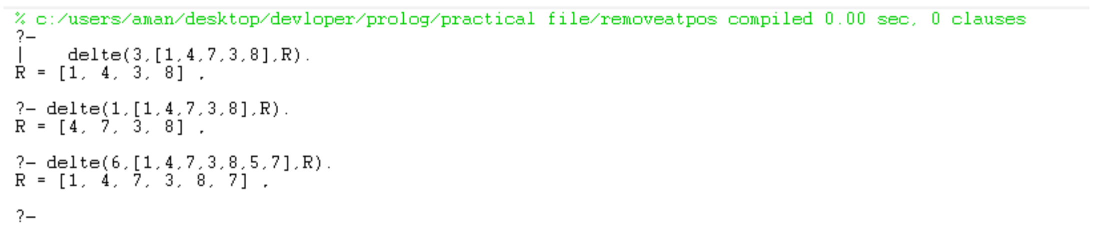
Knowledge Base:

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

delte(P,[X|Y],[X|R]):- P1 is P-1,

delte(P1,Y,R).

Output:



```
% c:/users/aman/desktop/devloper/prolog/practical file/removeatpos compiled 0.00 sec, 0 clauses
?-
|    delte(3,[1,4,7,3,8],R).
R = [1, 4, 3, 8] .

?- delte(1,[1,4,7,3,8],R).
R = [4, 7, 3, 8] .

?- delte(6,[1,4,7,3,8,5,7],R).
R = [1, 4, 7, 3, 8, 7] .

?-
```

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.

Knowledge Base:

dmerge([],L2,L2).

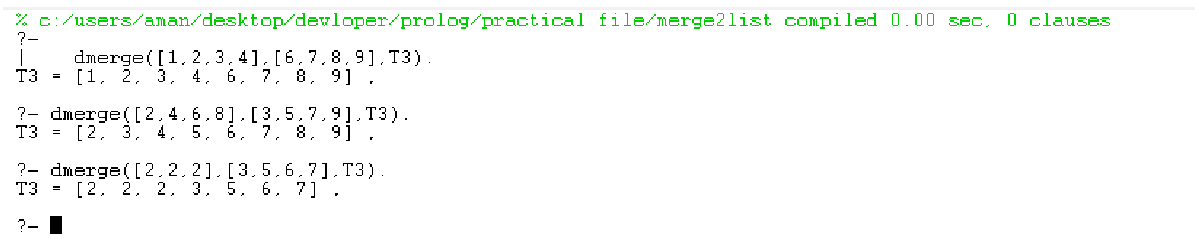
dmerge(L1,[],L1).

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).

Output:



```
% c:/users/aman/desktop/devloper/prolog/practical file/merge2list compiled 0.00 sec, 0 clauses
?-
|    dmerge([1,2,3,4],[6,7,8,9],T3).
T3 = [1, 2, 3, 4, 6, 7, 8, 9] .

?- dmerge([2,4,6,8],[3,5,7,9],T3).
T3 = [2, 3, 4, 5, 6, 7, 8, 9] .

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

?- 
```