Name	:	Nishika Chauhan
College Roll No.	:	21/5021
Email id	:	nishi12chauhann@gmail.com
Phone No.	:	9911907516
Paper Name	:	Artificial Intelligence
Paper Code	:	32341601
Type of Work	:	Practical File
Semester	:	VI
Submitted to	:	

Mr. Manish Kumar Singh

(Asst. Prof., Dept. of CS, SPM College For Women)

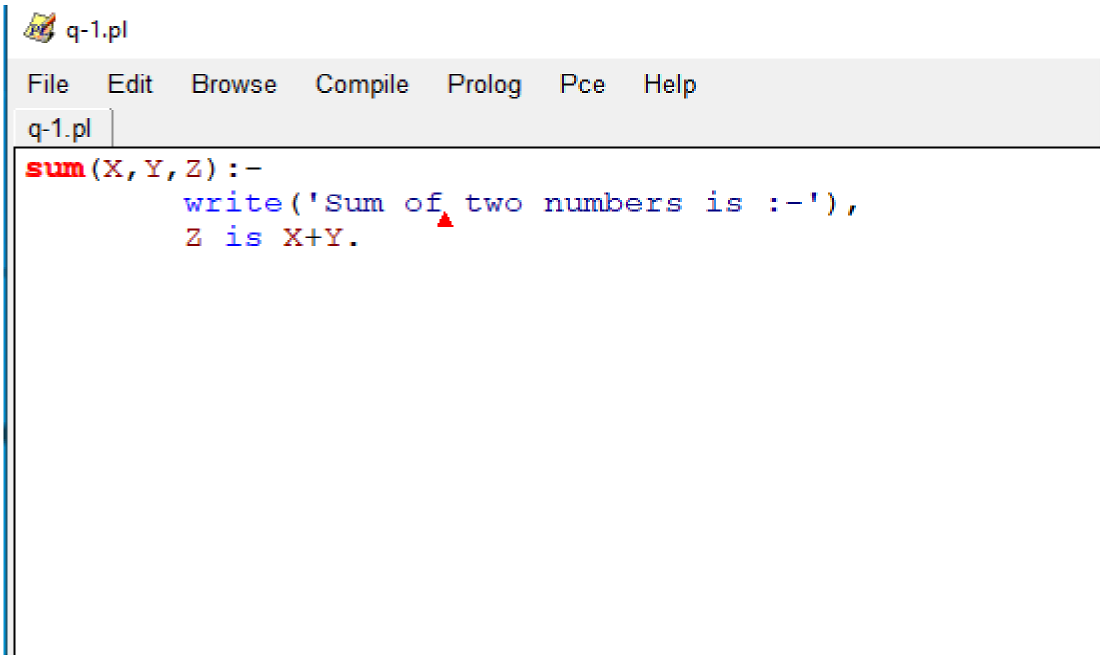
INDEX

12.	Write a Prolog program to implement sumlist(L., S) so that S is the sum of a given list L.	
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.	
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	
15.	Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.	
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.	
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.	
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.	

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

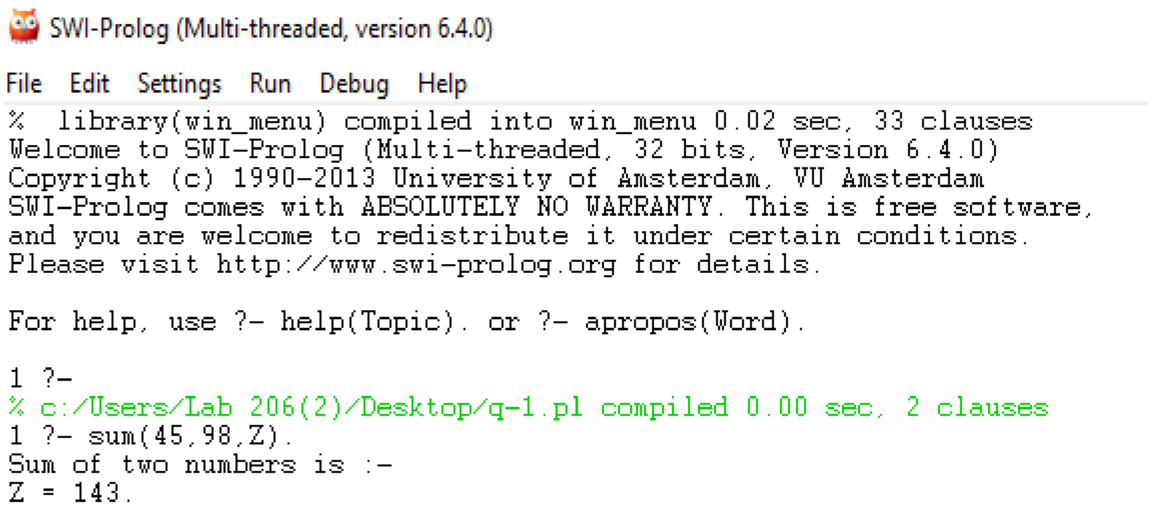
Code:

```
sum(X,Y,Z):-
        write('Sum of two numbers is :-') ,
         Z is X+Y .
```



```
sum(X,Y,Z):-
        write('Sum of two numbers is :-'),
        Z is X+Y.
```

Output :

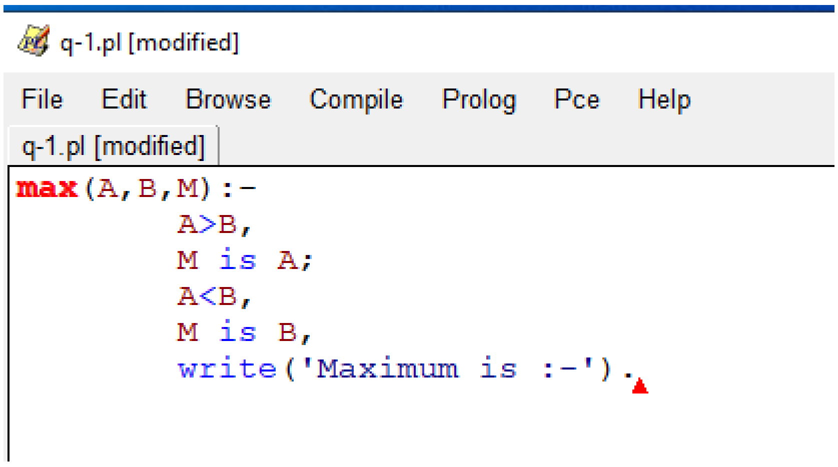

SWI-Prolog (Multi-threaded, version 6.4.0)
File Edit Settings Run Debug Help

```
%  library(win_menu) compiled into win_menu 0.02 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2 clauses
1 ?- sum(45,98,Z).
Sum of two numbers is :-
Z = 143.
```

Ques-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(A,B,M):-
        A>B,
        M is A;
        A<B,
        M is B,
        write(‘ Maximum is :- ‘).
```



```
max(A,B,M):-
        A>B,
        M is A;
        A<B,
        M is B,
        write('Maximum is :-').
```

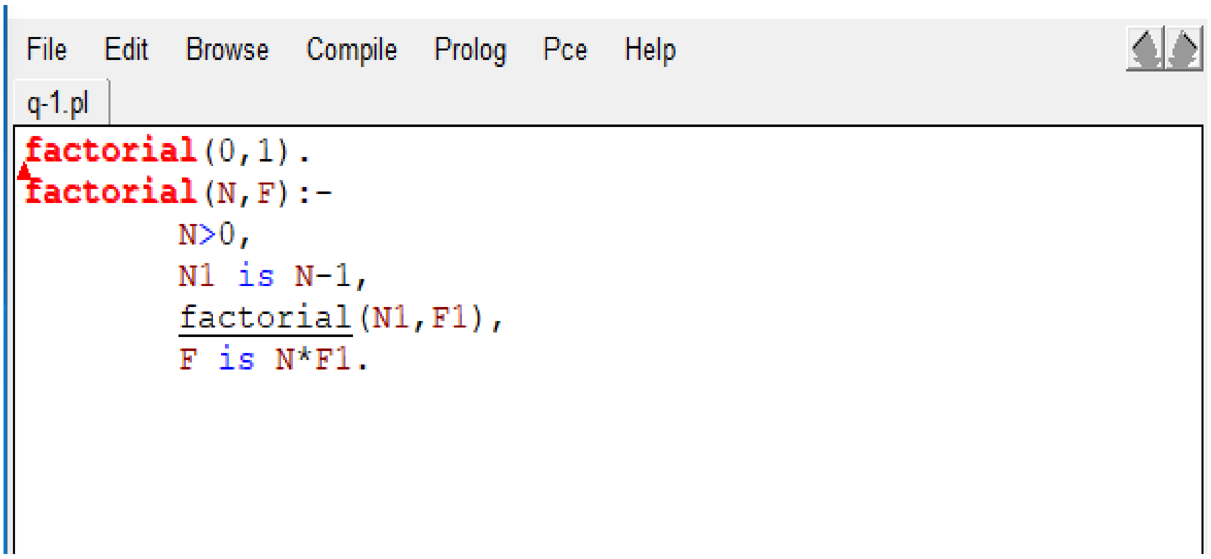
Output:

```
 2 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2 clauses
 2 ?- max(39,90,M).
Maximum is :-90
M = 90.

  3 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2 clauses
  3 ?- max(39,90,M).
Maximum is :-
M = 90.

 4 ?-
```

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

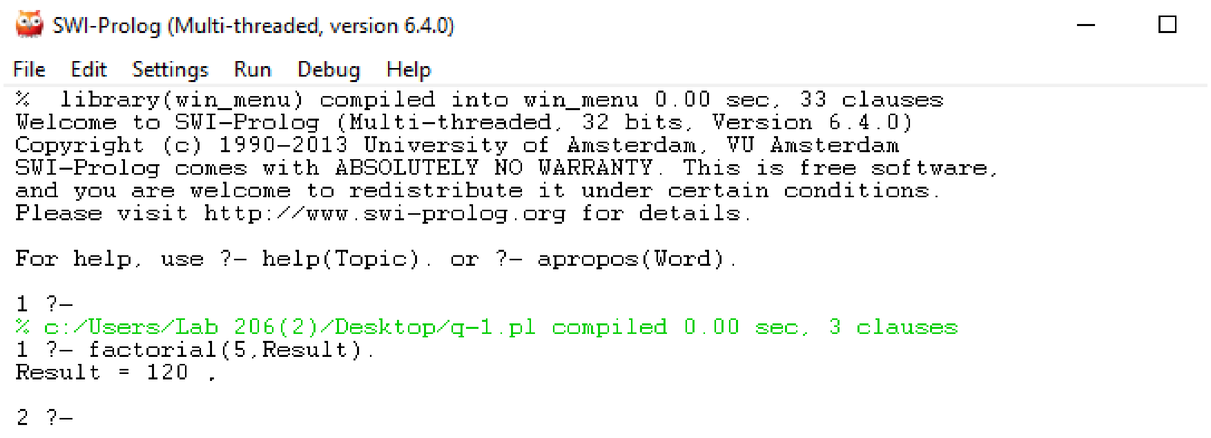
Code :

```
factorial(0,1).
factorial(N,F):-
        N>0,
        N1 is N-1,
        factorial(N1, F1),
        F is N*F1.
```



```
factorial(0,1).
factorial(N,F):-
        N>0,
        N1 is N-1,
        factorial(N1,F1),
        F is N*F1.
```

Output:

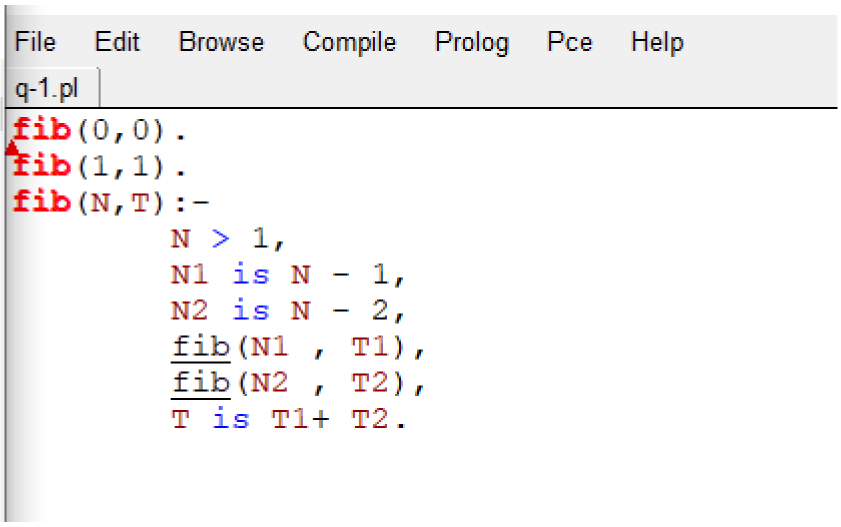


```
%  library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 3 clauses
1 ?- factorial(5,Result).
Result = 120 .

2 ?-
```

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

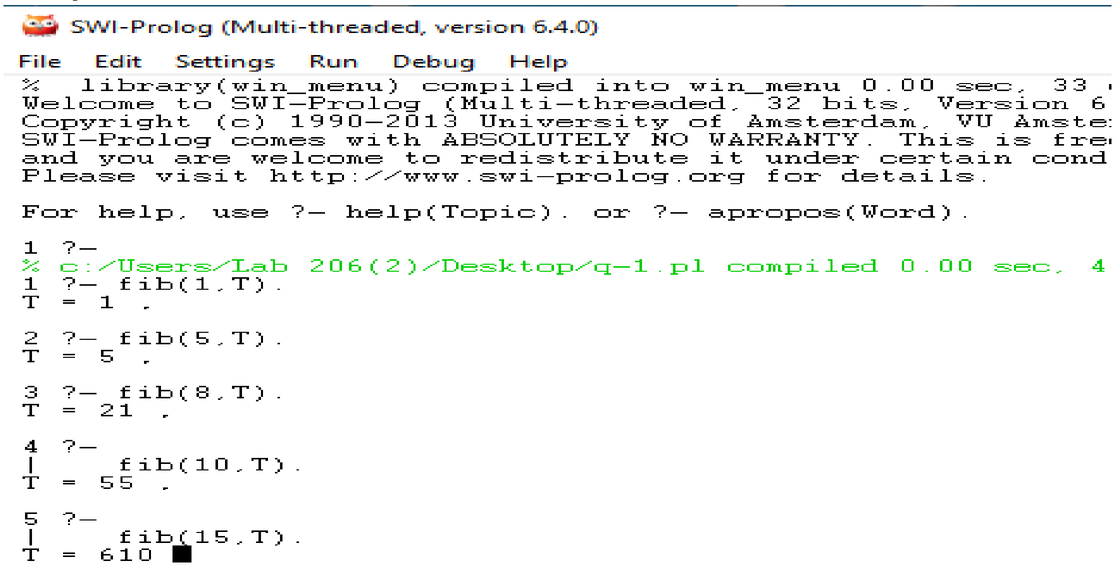
Code:

```
fib(0,0) .
fib(1,1) .
fib(N,T) :-
      N>1,
      N1 is N-1,
      N2 is N-2,
      fib(N1, T1),
      fib(N2 , T2),
      T is T1+T2.
```



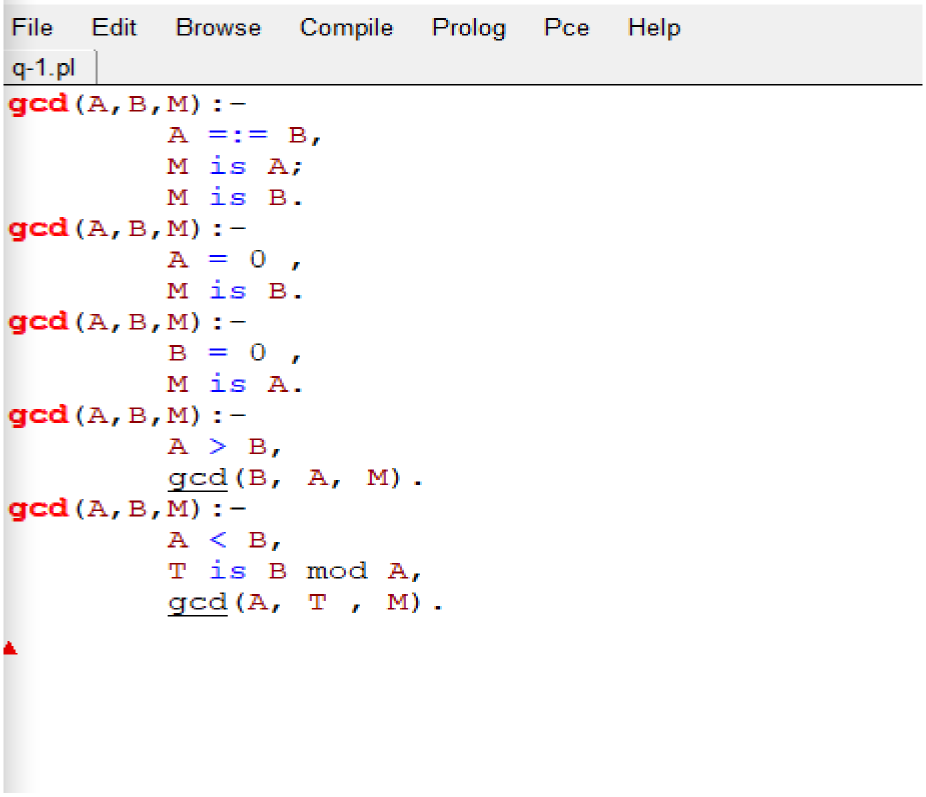
```
fib(0,0).
fib(1,1).
fib(N,T):-
        N > 1,
        N1 is N - 1,
        N2 is N - 2,
        fib(N1 , T1),
        fib(N2 , T2),
        T is T1+ T2.
```

Output:



```
% library(win_menu) compiled into win_menu 0.00 sec, 33
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6
Copyright (c) 1990-2013 University of Amsterdam, VU Amste
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is fre
and you are welcome to redistribute it under certain cond
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 4
1 ?- fib(1,T).
T = 1 .

2 ?- fib(5,T).
T = 5 .

3 ?- fib(8,T).
T = 21 .

4 ?-
|    fib(10,T).
T = 55 .

5 ?-
|    fib(15,T).
T = 610
```

Ques-5) Write a Prolog program to implement GCD of two numbers.
Code:



```
gcd(A,B,M):-
        A =:= B,
        M is A;
        M is B.
gcd(A,B,M):-
        A = 0 ,
        M is B.
gcd(A,B,M):-
        B = 0 ,
        M is A.
gcd(A,B,M):-
        A > B,
        gcd(B, A, M).
gcd(A,B,M):-
        A < B,
        T is B mod A,
        gcd(A, T , M).
```

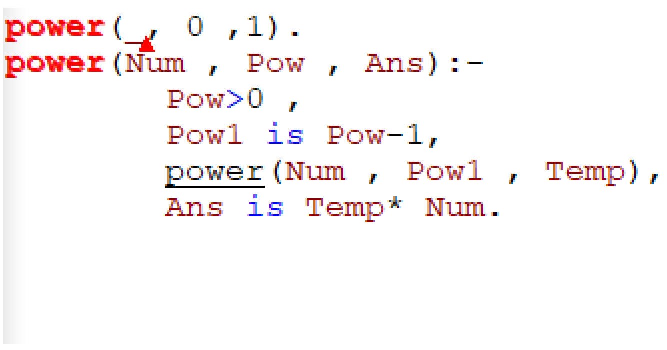
Output:

```
%  library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 6 clauses
1 ?- gcd(30,15,M).
M = 15 .

2 ?- gcd(15,30,M).
M = 30 .

3 ?- gcd(45,89,M).
M = 89 .

4 ?- gcd(90,68,M).
M = 68
```

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

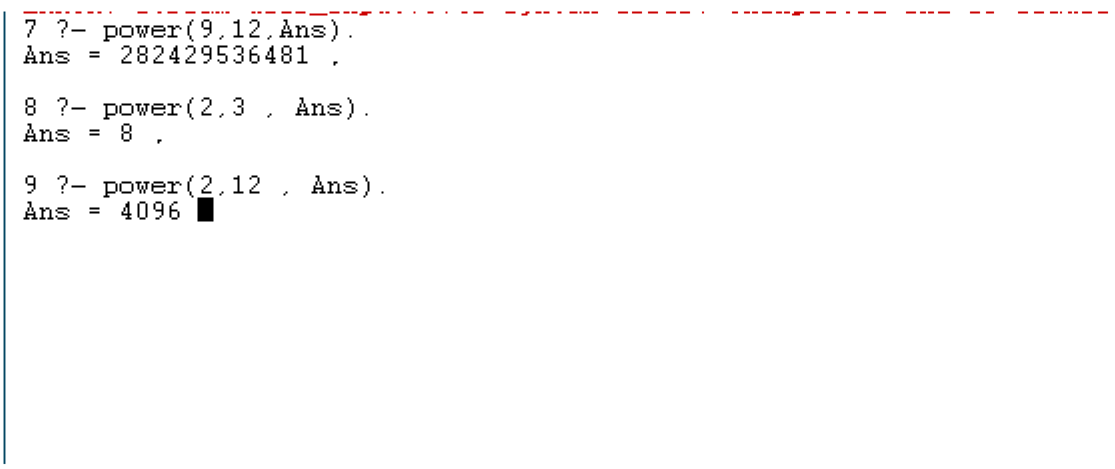
Code:

```
power(_ ,  0 , 1).
power(Num , Pow ,Ans):-
            Pow>0,
            Pow1 is Pow-1,
            power(Num , Pow1 , Temp),
            Ans is Temp*Num.
```



```
power(_,  0  ,1).
power(Num , Pow , Ans):-
        Pow>0 ,
        Pow1 is Pow-1,
        power(Num , Pow1 , Temp),
        Ans is Temp* Num.
```

Output:

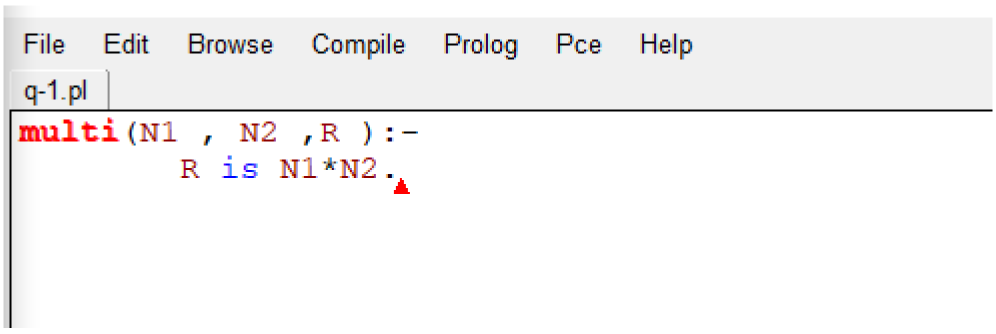


```
7 ?- power(9,12,Ans).
Ans = 282429536481 .

8 ?- power(2,3 , Ans).
Ans = 8 .

9 ?- power(2,12 , Ans).
Ans = 4096
```

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

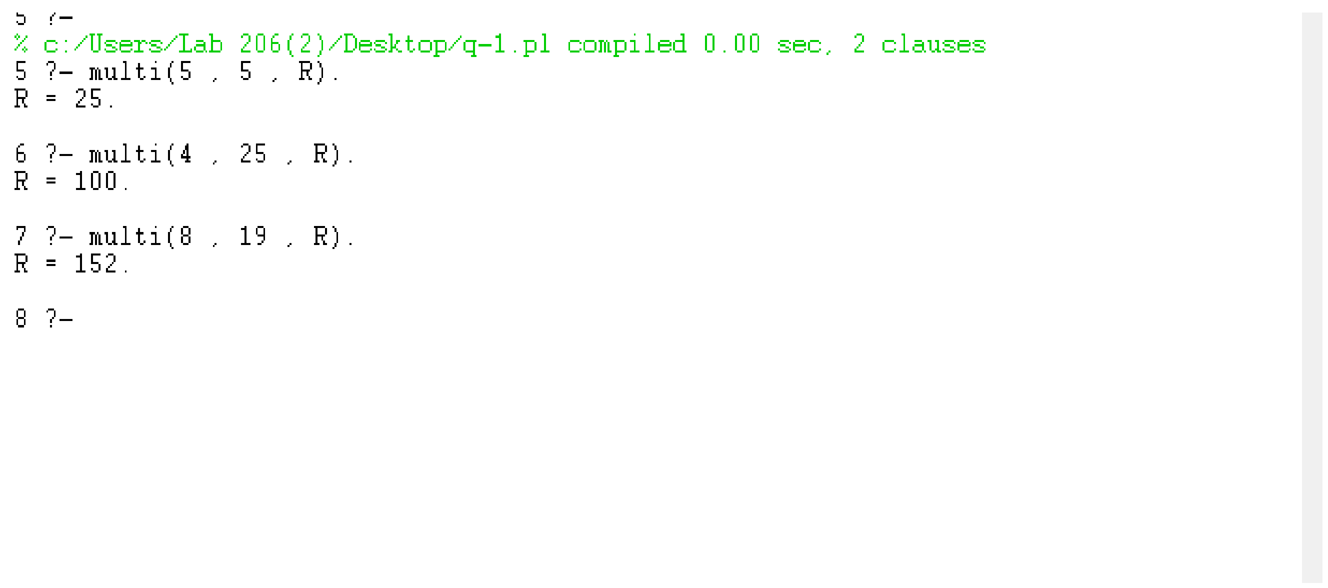
Code:

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



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

Output:



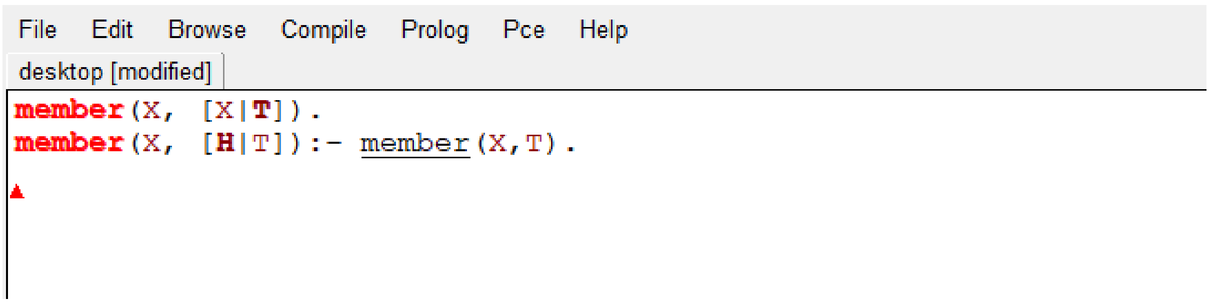
```
5 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2 clauses
5 ?- multi(5 , 5 , R).
R = 25.

6 ?- multi(4 , 25 , R).
R = 100.

7 ?- multi(8 , 19 , R).
R = 152.

8 ?-
```

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

Code:

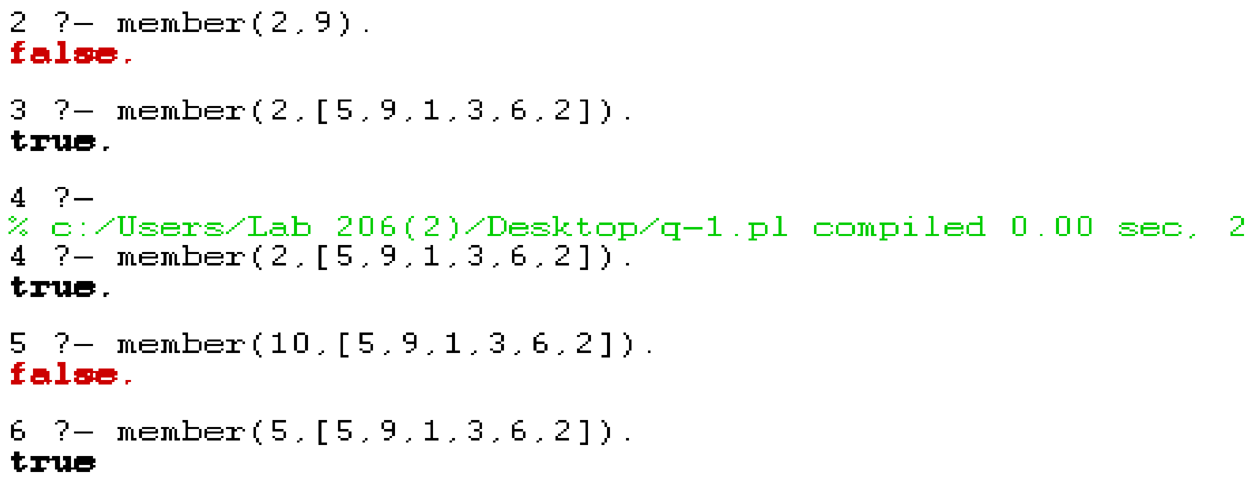
member(X , [X|T]).

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



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

Output:

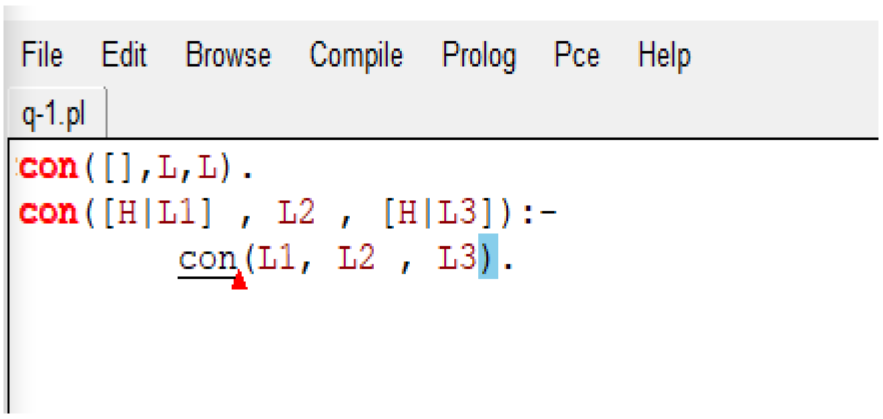


```
2 ?- member(2,9).
false.

3 ?- member(2,[5,9,1,3,6,2]).
true.

4 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2
4 ?- member(2,[5,9,1,3,6,2]).
true.

5 ?- member(10,[5,9,1,3,6,2]).
false.

6 ?- member(5,[5,9,1,3,6,2]).
true
```

Ques-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 L.3.

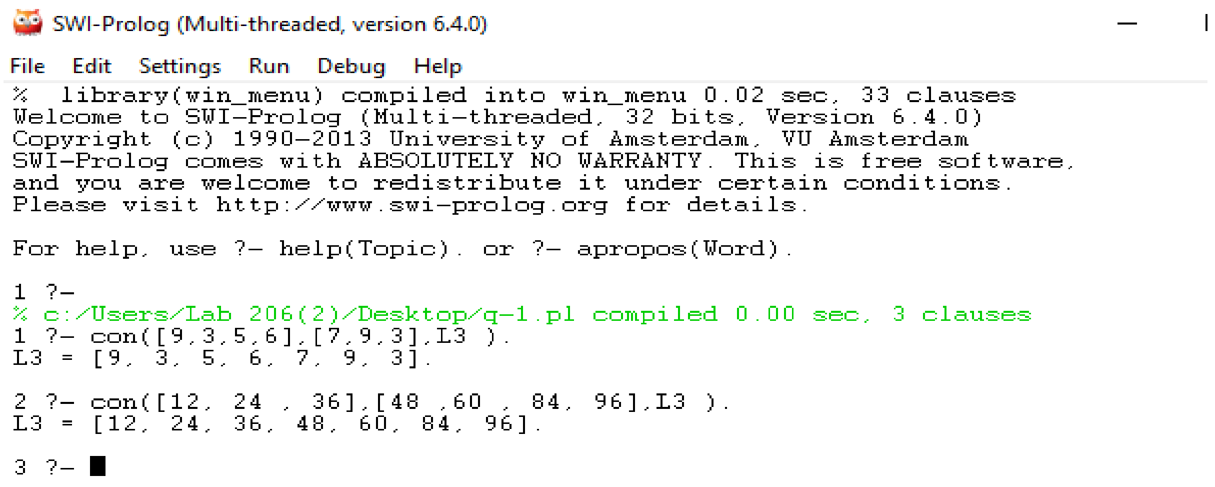
Code:

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

con([H|L1] , L2 , [H|L3]):-

        con(L1, L2 , L3).
```



```
con([],L,L).
con([H|L1] , L2 , [H|L3]):-
        con(L1, L2 , L3).
```

Output:

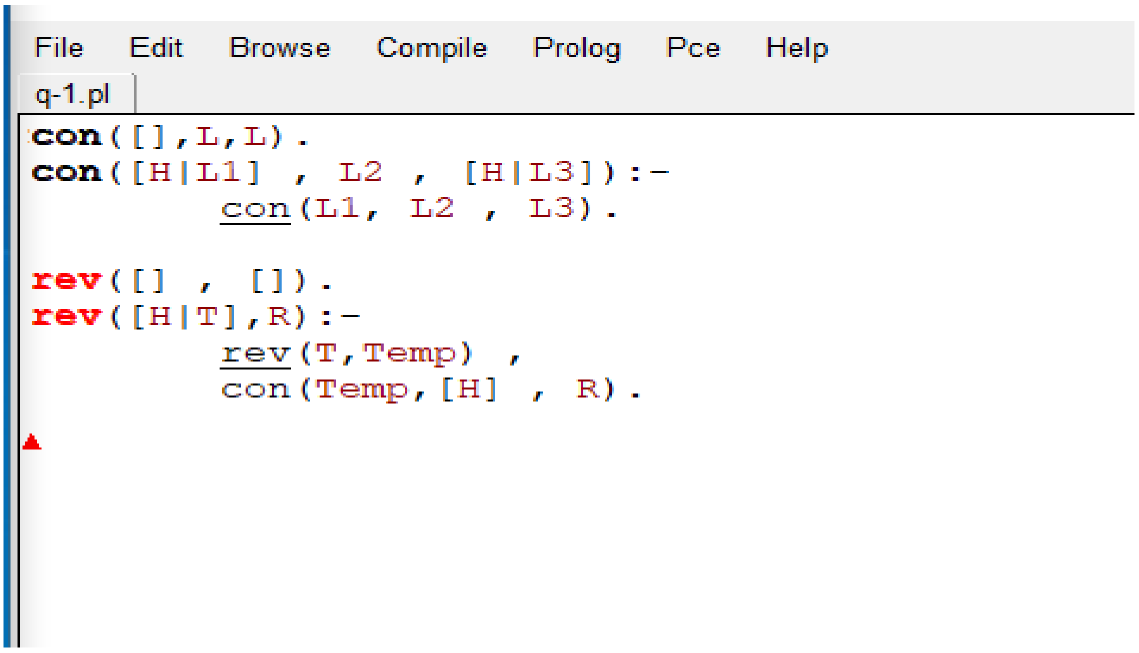


```
%  library(win_menu) compiled into win_menu 0.02 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 3 clauses
1 ?- con([9,3,5,6],[7,9,3],L3 ).
L3 = [9, 3, 5, 6, 7, 9, 3].

2 ?- con([12, 24 , 36],[48 ,60 , 84, 96],L3 ).
L3 = [12, 24, 36, 48, 60, 84, 96].

3 ?-
```

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

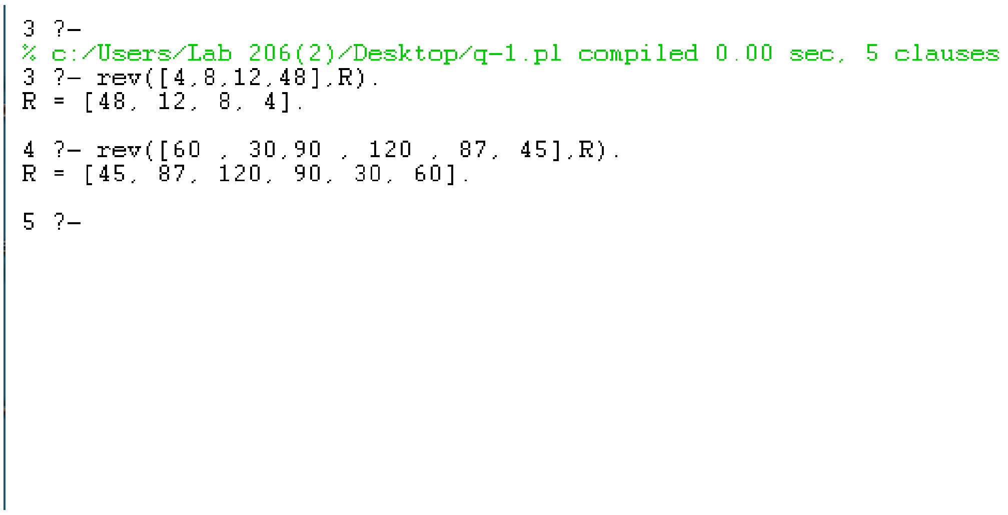
Code:

```
con([ ] , L , L).
con([H|L1] , L2 , [H|L3]):-
        con(L1, L2 , L3).
rev([ ] ,  [ ]).
rev( [H|T] , R):-
         rev(T , Temp) ,
         con(Temp , [H] , R).
```



```
con([],L,L).
con([H|L1] , L2 , [H|L3]):-
        con(L1, L2 , L3).

rev([] , []).
rev([H|T],R):-
        rev(T,Temp) ,
        con(Temp,[H] , R).
```

Output:

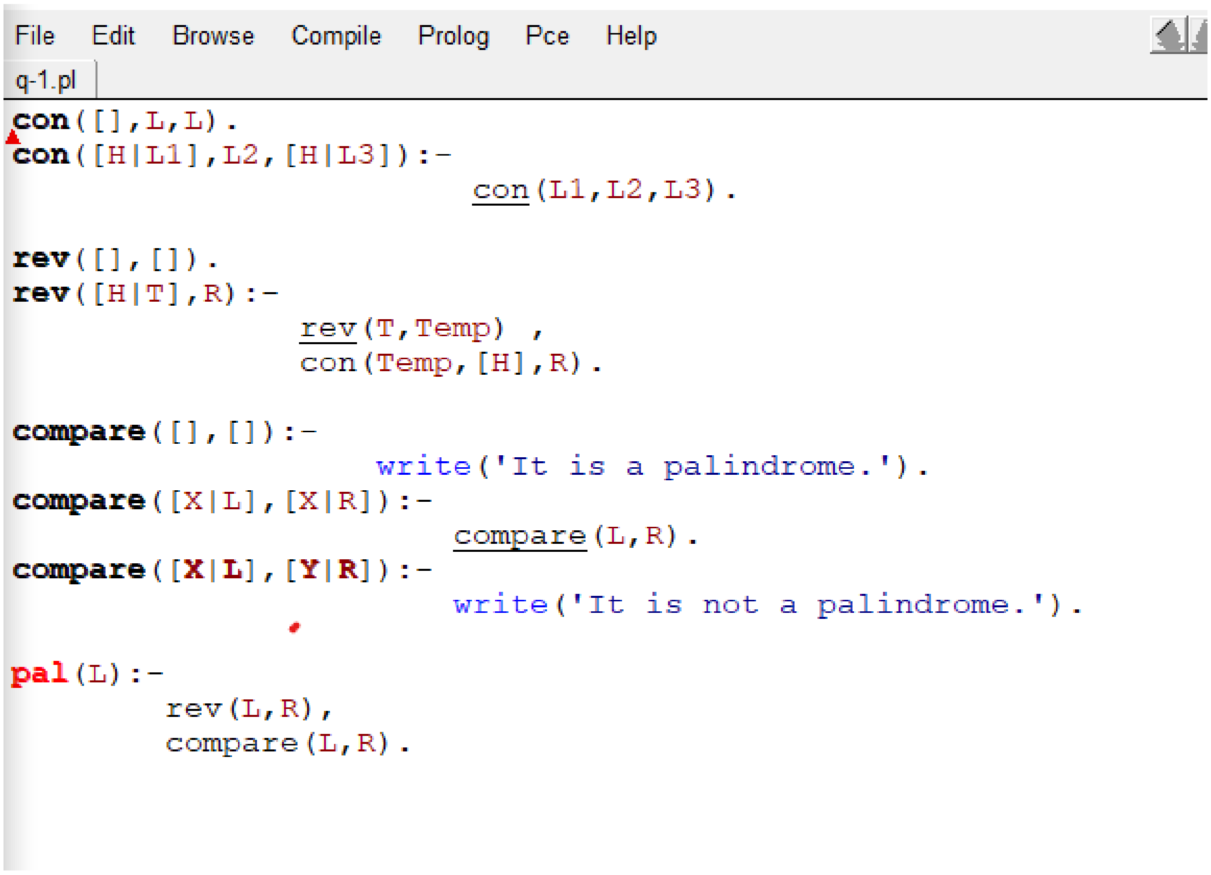


```
3 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 5 clauses
3 ?- rev([4,8,12,48],R).
R = [48, 12, 8, 4].

4 ?- rev([60 , 30,90 , 120 , 87, 45],R).
R = [45, 87, 120, 90, 30, 60].

5 ?-
```

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

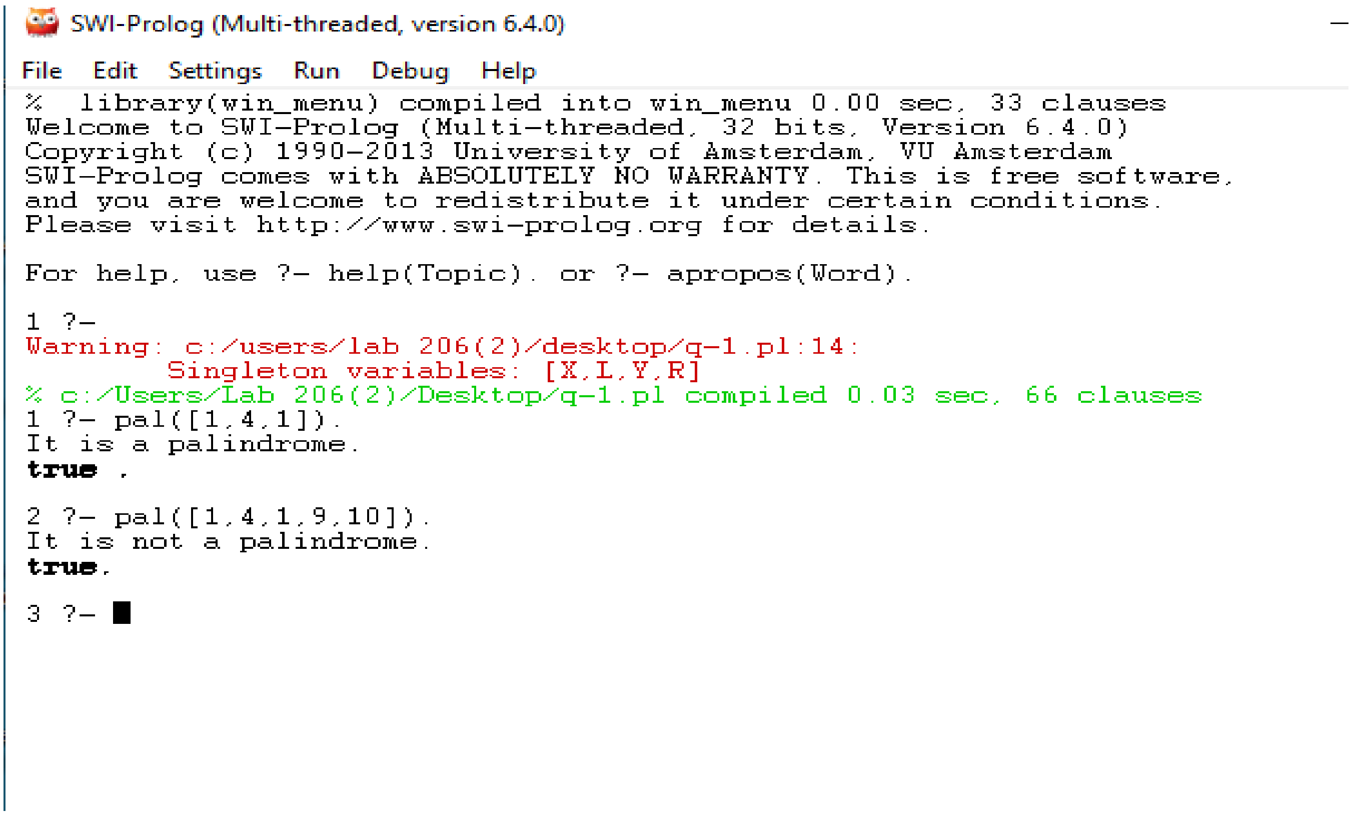
Code:

```
con([ ] , L , L).
con([H|L1] , L2 , [H|L3]):-
        con(L1, L2 , L3).
rev([ ] ,  [ ]).
rev( [H|T] , R):-
         rev(T , Temp) ,
         con(Temp , [H] , R).
compare([ ] , [ ]):-
              write('It s a palindrome').
compare( [X|L] , [X|R]):- compare(L, R).
compare([X|L] , [Y|R]):- write('It is not a palindrome.').
pal(L):-
      rev(L , R),
      compare(L , R).
```



```
con([],L,L).
con([H|L1],L2,[H|L3]):-
                     con(L1,L2,L3).

rev([],[]).
rev([H|T],R):-
              rev(T,Temp)  ,
              con(Temp,[H],R).

compare([],[]):-
                   write('It is a palindrome.').
compare([X|L],[X|R]):-
                        compare(L,R).
compare([X|L],[Y|R]):-
                        write('It is not a palindrome.').

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

Output:


SWI-Prolog (Multi-threaded, version 6.4.0)

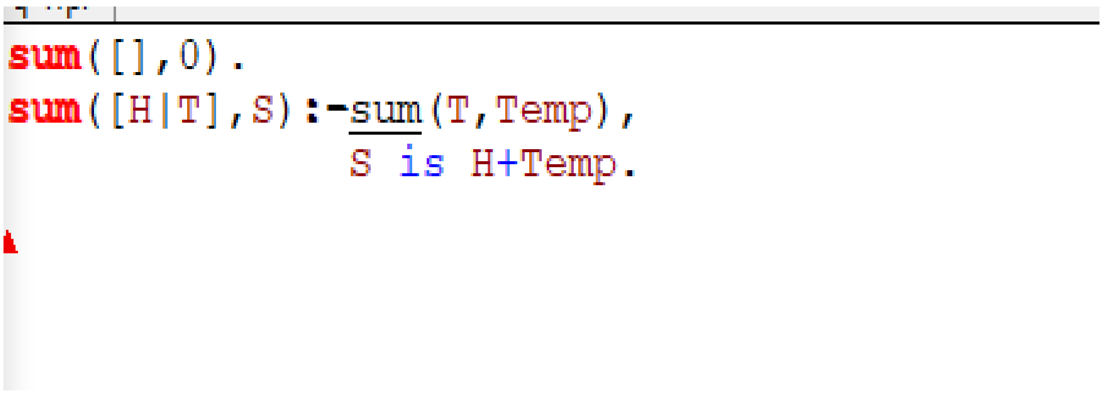
File Edit Settings Run Debug Help

```
%  library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
Warning: c:/users/lab 206(2)/desktop/q-1.pl:14:
        Singleton variables: [X,L,Y,R]
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.03 sec, 66 clauses
1 ?- pal([1,4,1]).
It is a palindrome.
true .

2 ?- pal([1,4,1,9,10]).
It is not a palindrome.
true.

3 ?- 
```

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

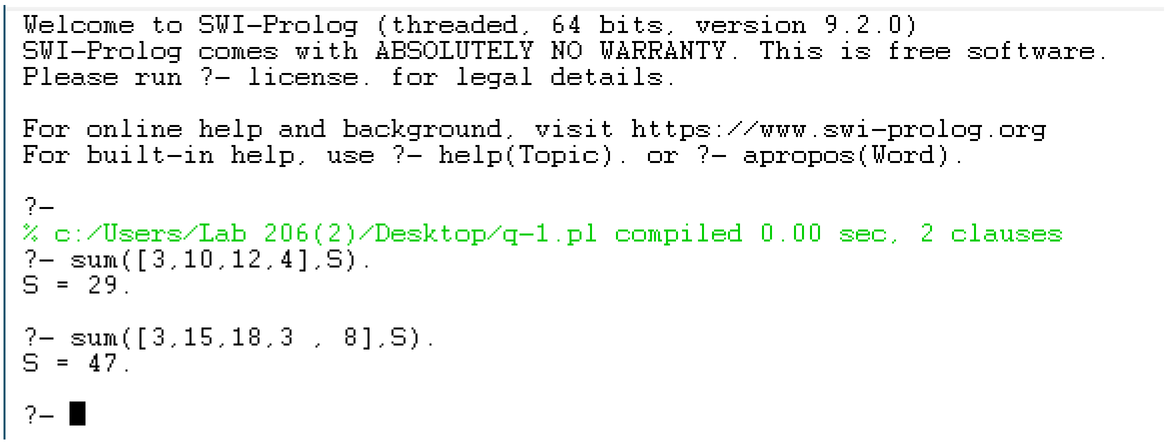
Code:

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



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

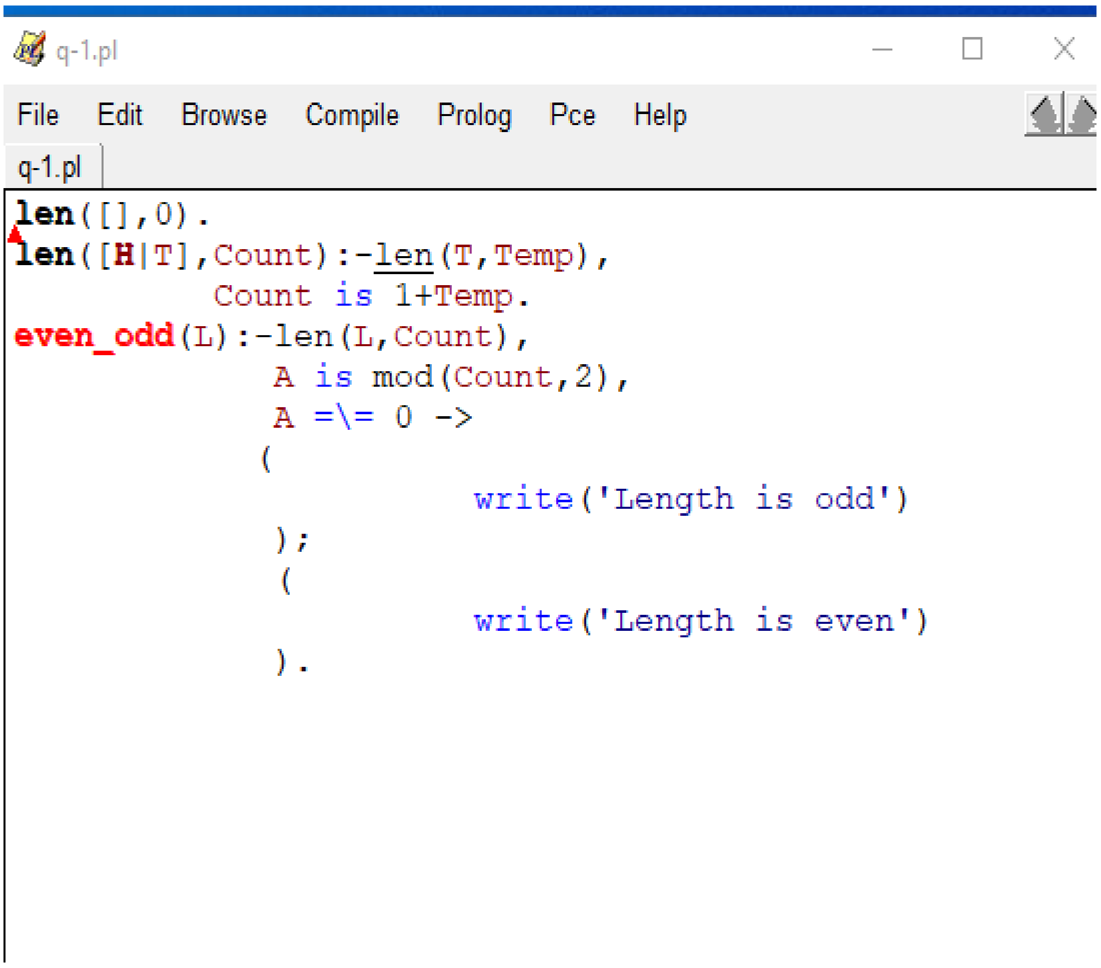
Output:



```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.2.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 2 clauses
?- sum([3,10,12,4],S).
S = 29.

?- sum([3,15,18,3 , 8],S).
S = 47.

?-
```

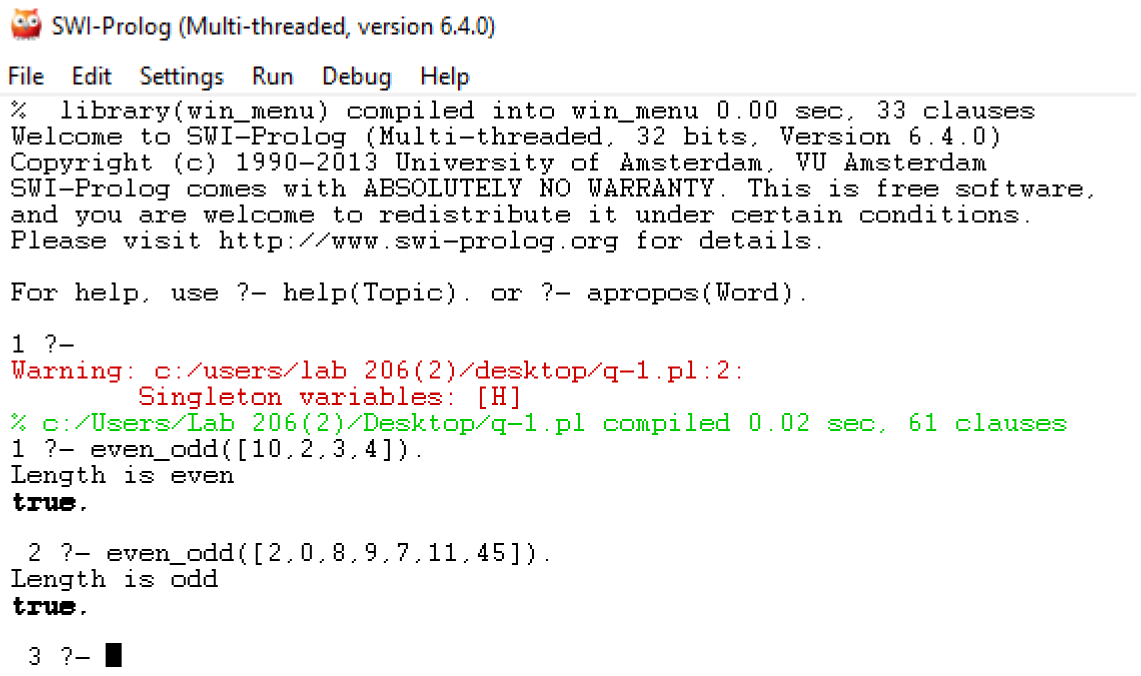
Ques-13) Write a Prolog program to implement two predicates evenlength(List) and oddlength(List) so that they are true if their argument is a list of even or odd length respectively.

Code:



```
len([],0).
len([H|T],Count):-len(T,Temp),
            Count is 1+Temp.
even_odd(L):-len(L,Count),
               A is mod(Count,2),
               A =\= 0 ->
              (
                         write('Length is odd')
               );
               (
                         write('Length is even')
               ).
```

Output:


SWI-Prolog (Multi-threaded, version 6.4.0)
File Edit Settings Run Debug Help

```
%  library(win_menu) compiled into win_menu 0.00 sec, 33 clauses
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 6.4.0)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?-
Warning: c:/users/lab 206(2)/desktop/q-1.pl:2:
        Singleton variables: [H]
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.02 sec, 61 clauses
1 ?- even_odd([10,2,3,4]).
Length is even
true.

 2 ?- even_odd([2,0,8,9,7,11,45]).
Length is odd
true.

 3 ?-
```

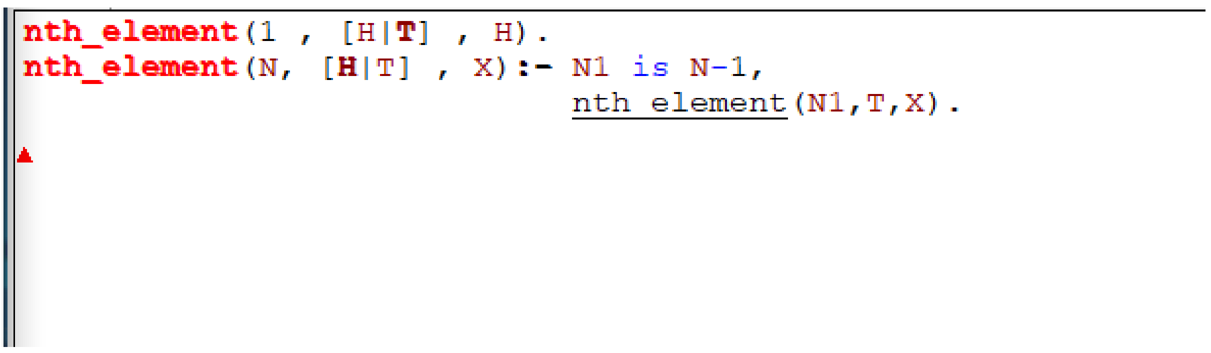
Ques-14) Write a Prolog program to implement nth_element (N, L, X) where N is the desired position, L is a list and X represents the Nth element of L.

Code:

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

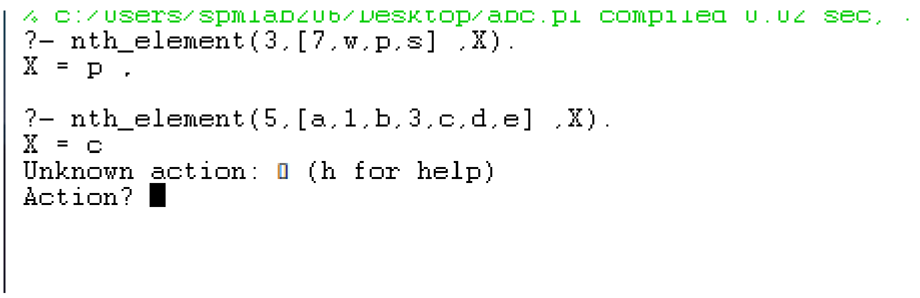
nth_element(N, [H|T] , X):- N1 is N-1,

nth element(N1 , T , X).



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

Output:

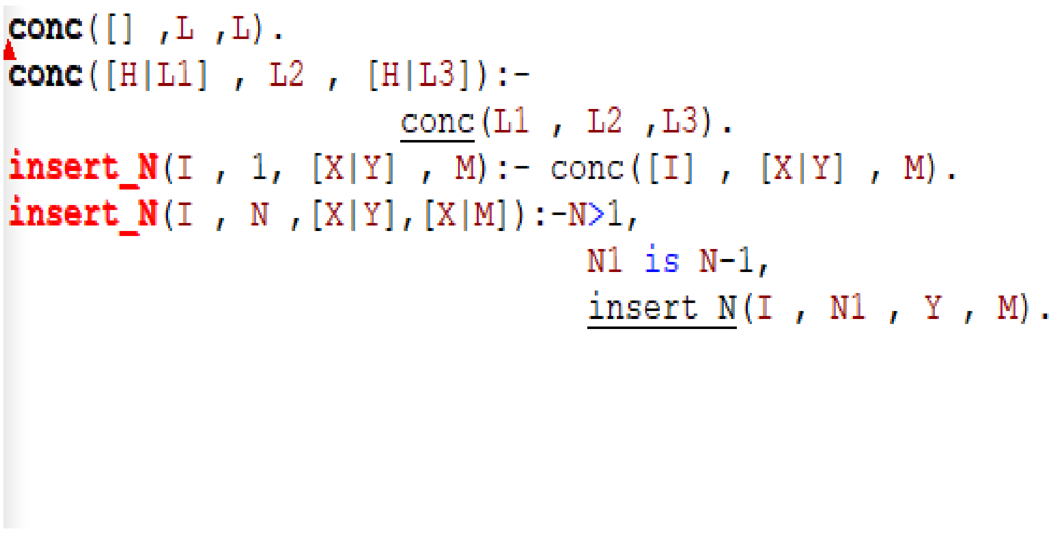


```
% c:/users/spmlab206/desktop/abc.pl compiled 0.02 sec, .
?- nth_element(3,[7,w,p,s] ,X).
X = p ,

?- nth_element(5,[a,1,b,3,c,d,e] ,X).
X = c
Unknown action: ▯ (h for help)
Action? █
```

Ques-15) 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.

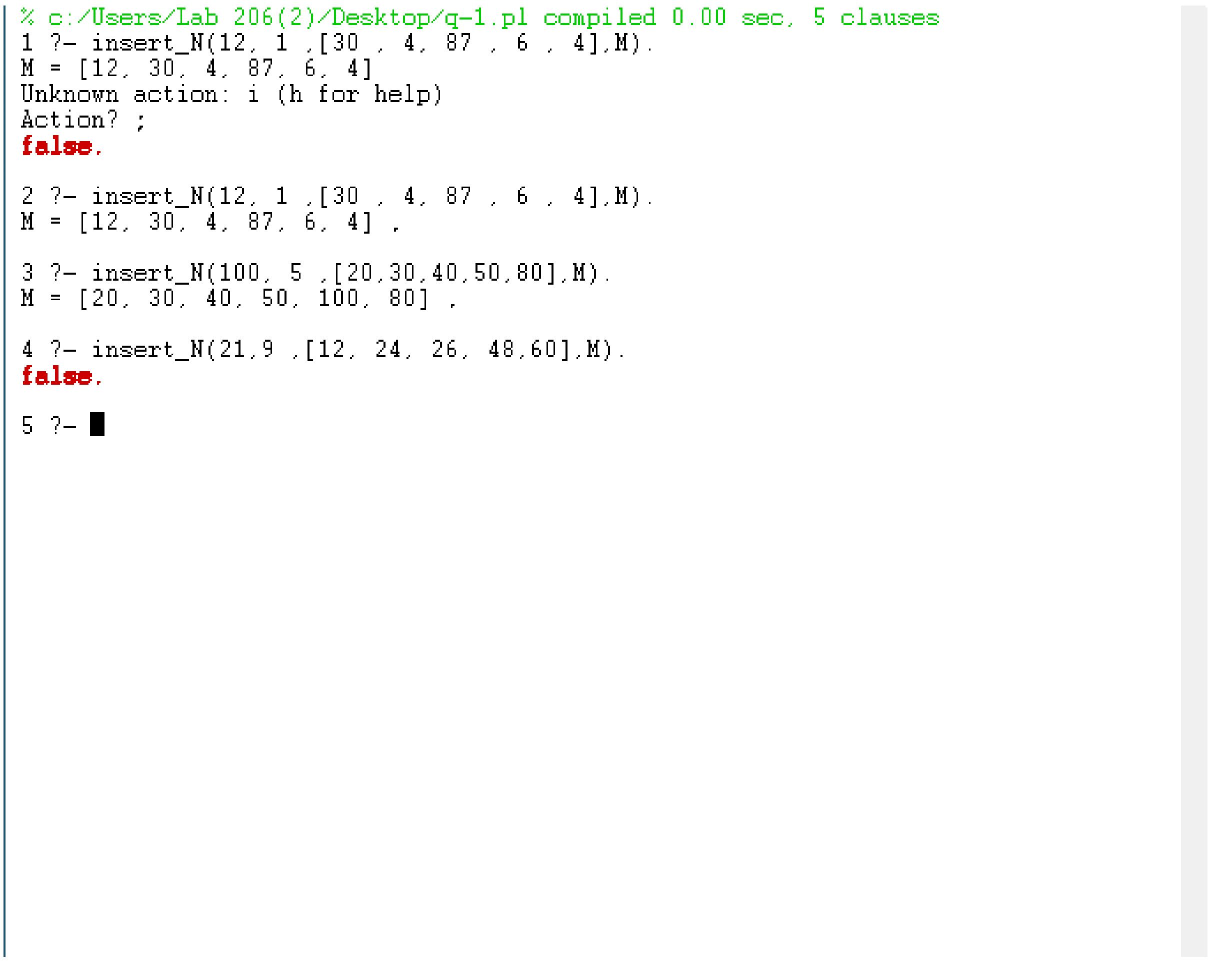
Code:

```
conc([] , L, L).
conc([H|L1] , L2 , [H|L3]):-
                              conc(L1 , L2 , L3).
insert_N(I , 1 , [X|Y] , M):- conc([I] , [X|Y], , M).
insert_N(I , N , [X|Y]) : - N>1 ,
                              N1 is N-1,
                              insert_N(I , N1 , Y, M).
```



```
conc([] ,L ,L).
conc([H|L1] , L2 , [H|L3]):-
                    conc(L1 , L2 ,L3).
insert_N(I , 1, [X|Y] , M):- conc([I] , [X|Y] , M).
insert_N(I , N ,[X|Y],[X|M]):-N>1,
                              N1 is N-1,
                              insert_N(I , N1 , Y , M).
```

Output:



```
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 5 clauses
1 ?- insert_N(12, 1 ,[30 , 4, 87 , 6 , 4],M).
M = [12, 30, 4, 87, 6, 4]
Unknown action: i (h for help)
Action? ;
false.

2 ?- insert_N(12, 1 ,[30 , 4, 87 , 6 , 4],M).
M = [12, 30, 4, 87, 6, 4] .

3 ?- insert_N(100, 5 ,[20,30,40,50,80],M).
M = [20, 30, 40, 50, 100, 80] .

4 ?- insert_N(21,9 ,[12, 24, 26, 48,60],M).
false.

5 ?- 
```

Ques-16) Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.

Code:

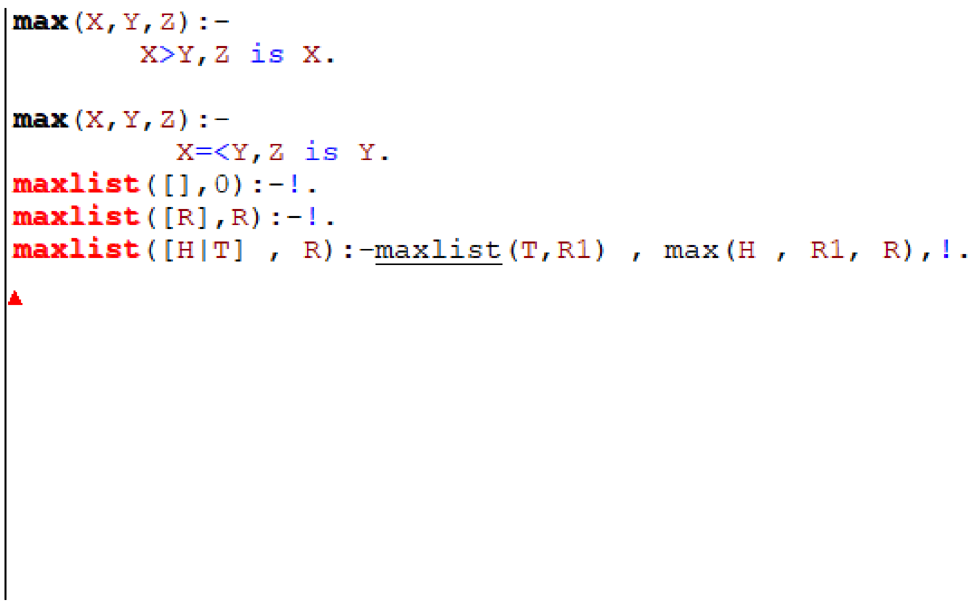
max(X,Y,Z):- X>Y,Z is X.

max(X,Y,Z):- X =< Y , Z s Y.

maxlist([] , 0) :- !.

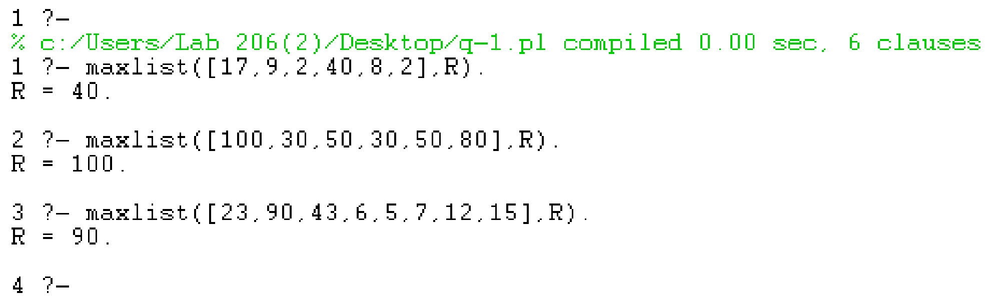
maxlist([R] , R) :- !.

maxlist([H|T] , R) :- maxlist(T ,R1) , max(H , R1 , R) , !.



```
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 :

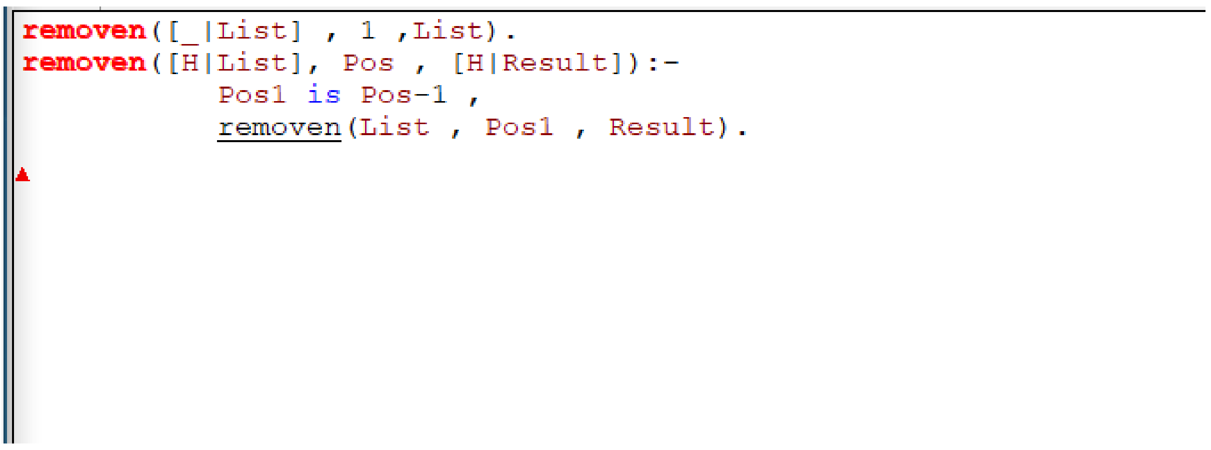


```
1 ?-
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 6 clauses
1 ?- maxlist([17,9,2,40,8,2],R).
R = 40.

2 ?- maxlist([100,30,50,30,50,80],R).
R = 100.

3 ?- maxlist([23,90,43,6,5,7,12,15],R).
R = 90.

4 ?-
```

Ques-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.

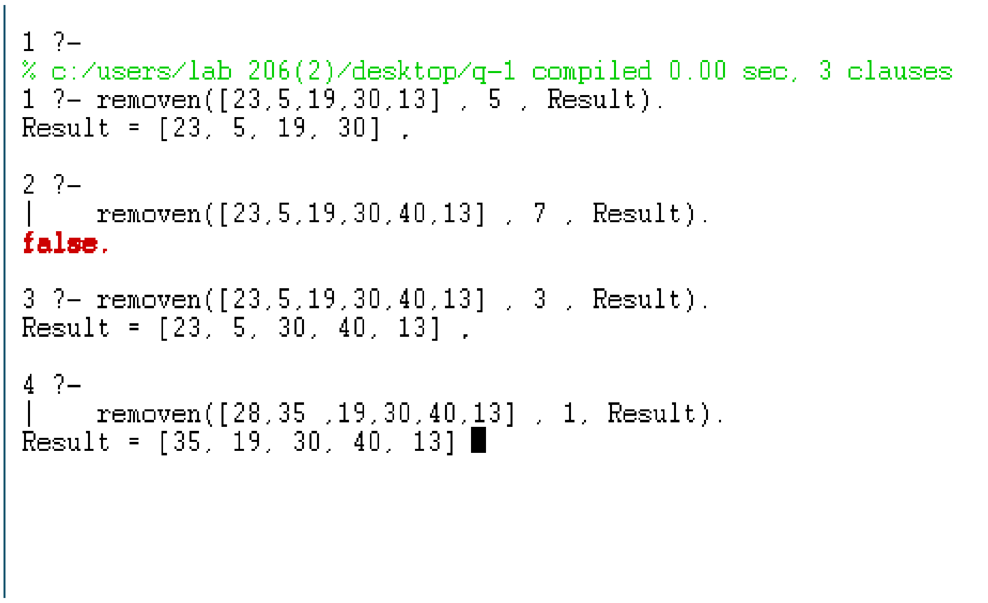
Code:

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



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

Output:

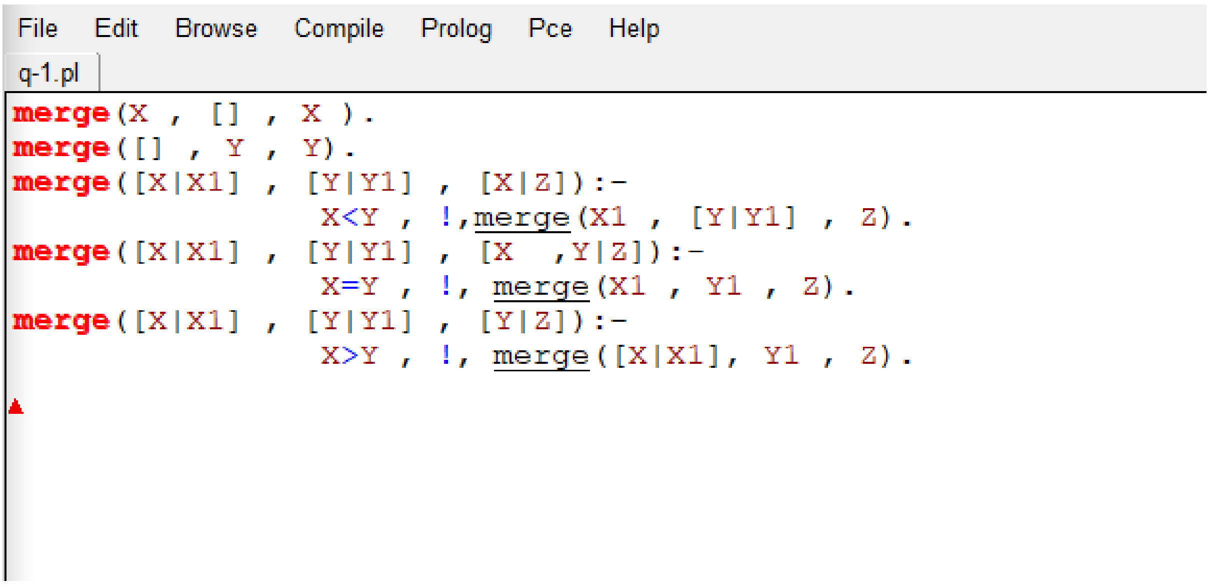


```
1 ?-
% c:/users/lab 206(2)/desktop/q-1 compiled 0.00 sec, 3 clauses
1 ?- removen([23,5,19,30,13] , 5 , Result).
Result = [23, 5, 19, 30] .

2 ?-
|    removen([23,5,19,30,40,13] , 7 , Result).
false.

3 ?- removen([23,5,19,30,40,13] , 3 , Result).
Result = [23, 5, 30, 40, 13] .

4 ?-
|    removen([28,35 ,19,30,40,13] , 1, Result).
Result = [35, 19, 30, 40, 13]
```

Ques-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.

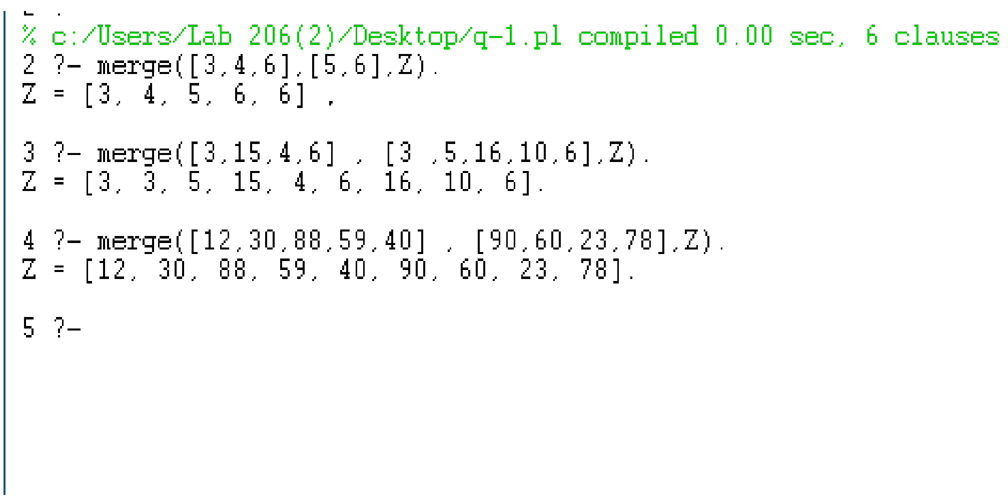
Code:

```
merge(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).
```



```
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:



```
% c:/Users/Lab 206(2)/Desktop/q-1.pl compiled 0.00 sec, 6 clauses
2 ?- merge([3,4,6],[5,6],Z).
Z = [3, 4, 5, 6, 6] .

3 ?- merge([3,15,4,6] , [3 ,5,16,10,6],Z).
Z = [3, 3, 5, 15, 4, 6, 16, 10, 6].

4 ?- merge([12,30,88,59,40] , [90,60,23,78],Z).
Z = [12, 30, 88, 59, 40, 90, 60, 23, 78].

5 ?-
```