

Artificial Intelligence Practicals

May 12, 2020

Submitted by:-

Name: Rohit Sharma

Roll no: 5781

Email Id: rohitsdec4@gmail.com

Phone No.: 8700486627

Course: B.sc(H) Computer Science, III Year, Section-B

This document consists of the Artificial Intelligence practicals along with their output.

Contents

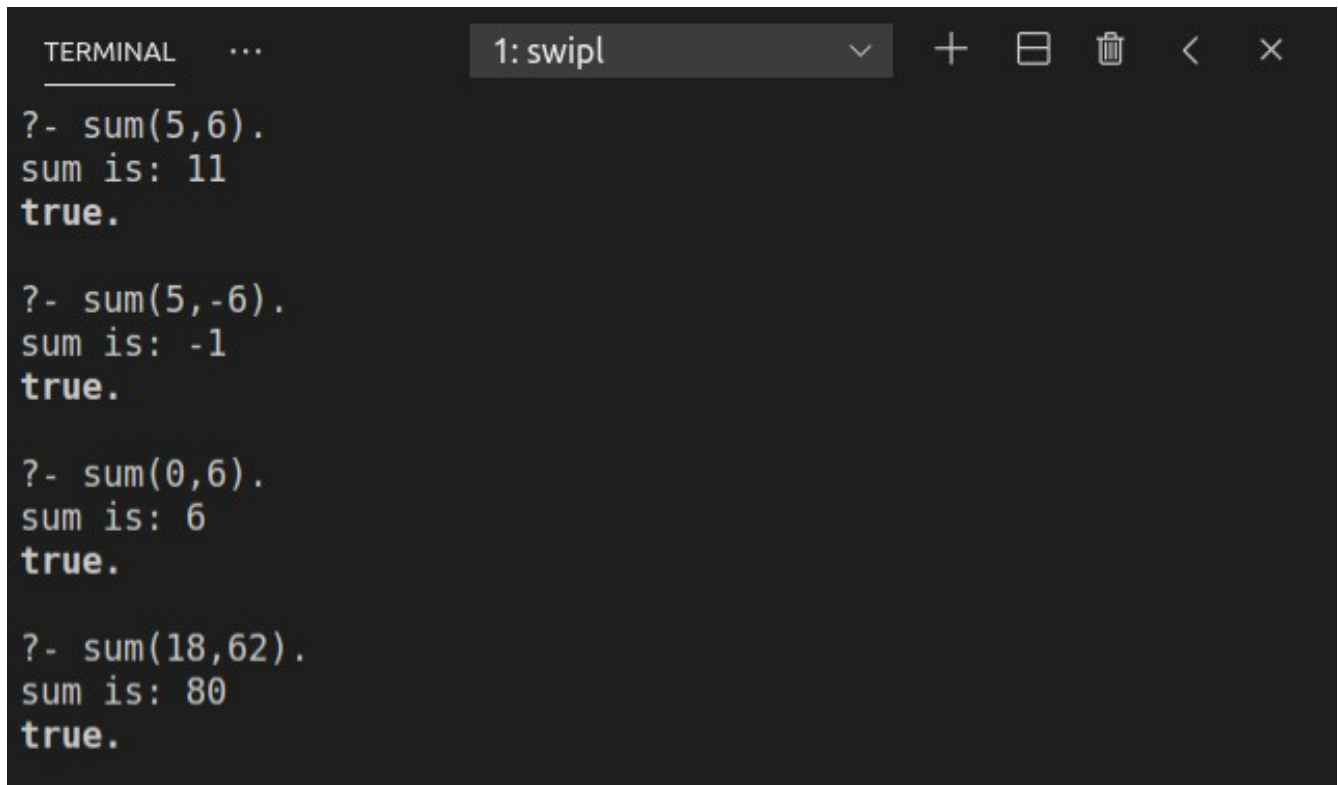
This document consists of the Artificial Intelligence practicals along with their output.	1
.....	
Question 1	3
Question 2	4
Question 3	5
Question 4	6
Question 5	7
Question 6	8
Question 7	9
Question 8	10
Question 9	11
Question 10	12
Question 11	13
Question 12	14
Question 13	15
Question 14	16
Question 15	17
Question 16	18
Question 17	19
Question 18	20
Question 19	21
Question 20	22
Question 21	23
Question 22	24
Question 23	25

Question 1

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

Solution 1

```
sum(X,Y):-  
(  
    R is X + Y,  
    write("Sum is: "),  
    write(R)  
).
```



```
1: swipl  
?- sum(5,6).  
sum is: 11  
true.  
  
?- sum(5,-6).  
sum is: -1  
true.  
  
?- sum(0,6).  
sum is: 6  
true.  
  
?- sum(18,62).  
sum is: 80  
true.
```

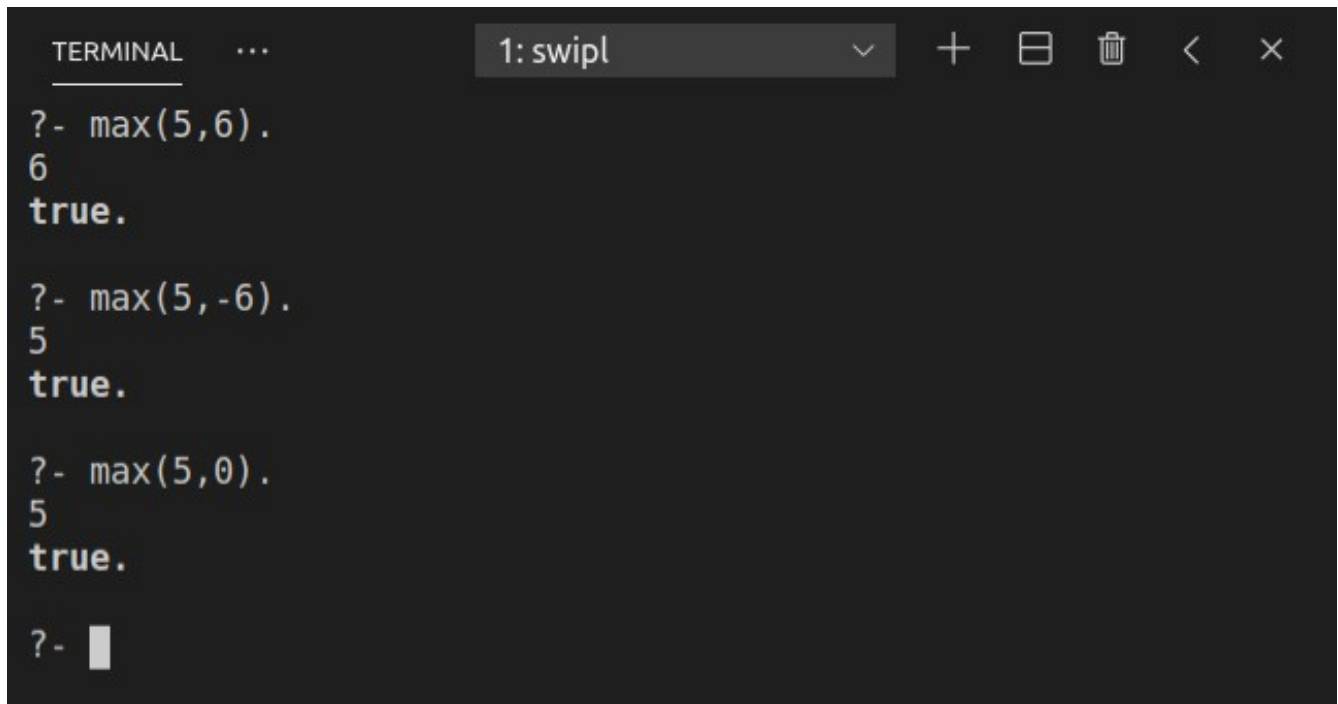
Figure 1: Solution 1 Image

Question 2

Write a prolog program to find maximum of 2 numbers.

Solution 2

```
max(X,Y):-  
(  
    X>Y  
    -> print(X)  
    ; print(Y)  
).
```



The image shows a terminal window with a dark background. The title bar at the top includes the word "TERMINAL", a menu icon, and a tab labeled "1: swipl". The terminal content shows three Prolog queries and their results:

```
?- max(5,6).  
6  
true.  
  
?- max(5,-6).  
5  
true.  
  
?- max(5,0).  
5  
true.  
  
?-   
|
```

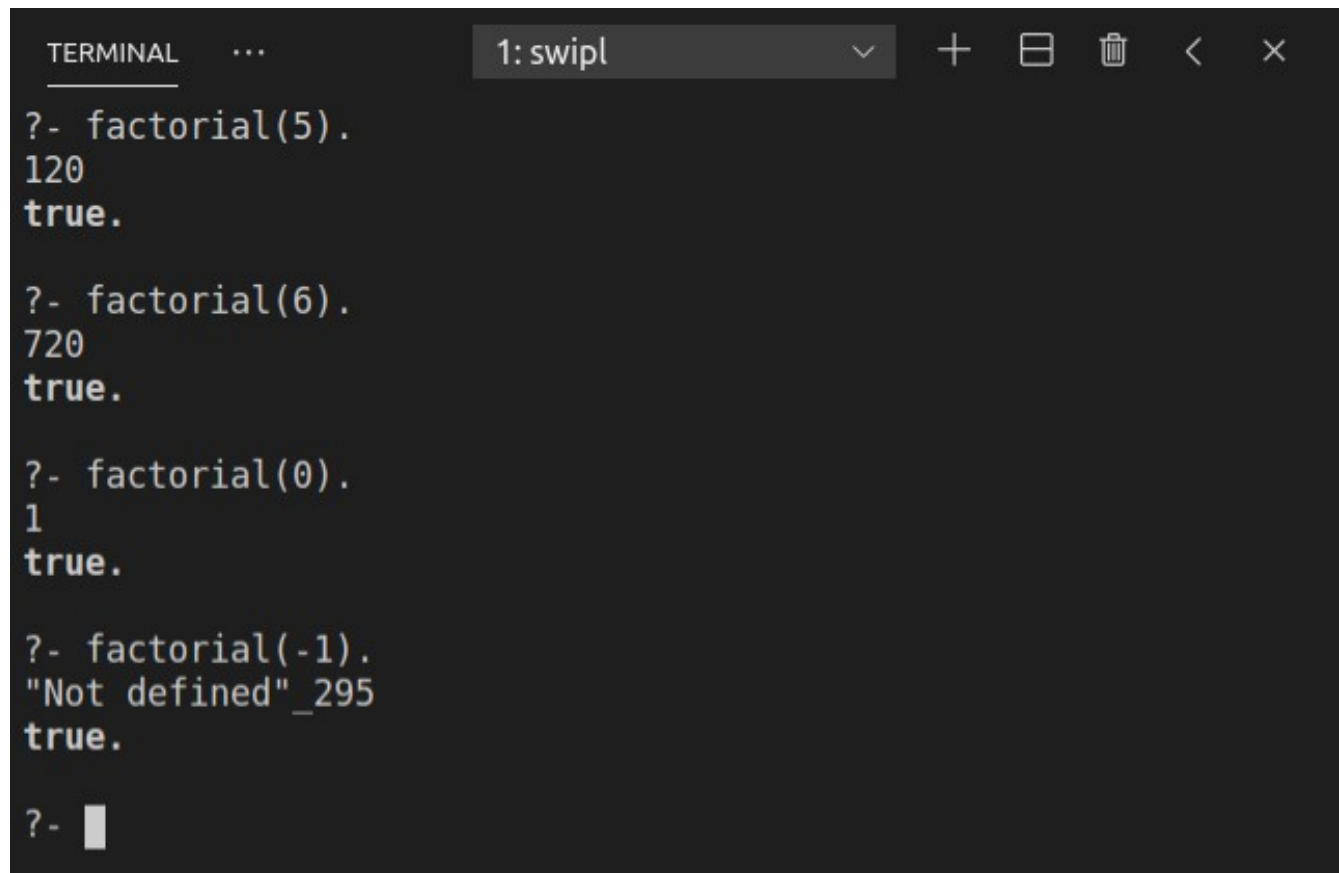
Figure 2: Solution 2 Image

Question 3

Write a prolog program to implement factorial of a number N.

Solution 3

```
factorial(N):- factorial(N, A), print(A),!.  
factorial(0,1).  
factorial(N,F):-  
(  
    N > 0  
    -> (  
        N1 is N-1,  
        factorial(N1, F1),  
        F is N*F1  
    )  
    ; print("Not defined"),!  
).
```



The image shows a terminal window with a dark background. The title bar at the top reads "1: swipl". The terminal content shows several Prolog queries and their results:

```
?- factorial(5).  
120  
true.  
  
?- factorial(6).  
720  
true.  
  
?- factorial(0).  
1  
true.  
  
?- factorial(-1).  
"Not defined"_295  
true.  
  
?-   
|
```

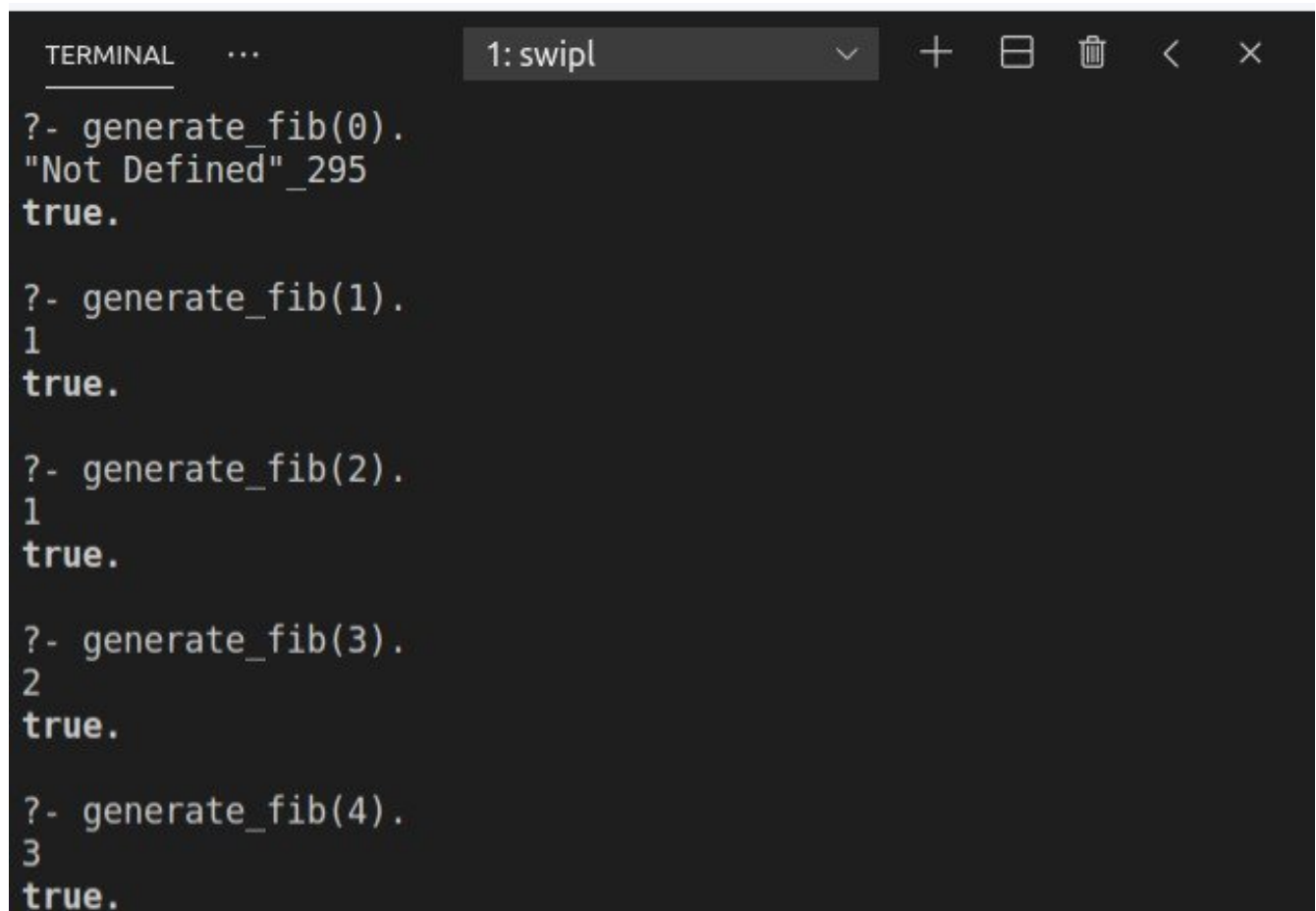
Figure 3: Solution 3 Image

Question 4

Write a prolog program to print the Nth term of the fibonacci series.

Solution 4

```
generate_fib(N):-generate_fib(N, A), print(A),!.
generate_fib(1,1).
generate_fib(2,1).
generate_fib(N,T):-
(
    N > 0
    ->(
        N1 is N-1,
        N2 is N-2,
        generate_fib(N1, T1),
        generate_fib(N2, T2),
        T is T1 + T2
    )
    ; print("Not Defined"),!
).
```



The screenshot shows a terminal window titled "1: swipl" with a dark background. The terminal displays the following Prolog queries and their results:

```
?- generate_fib(0).
"Not Defined"_295
true.

?- generate_fib(1).
1
true.

?- generate_fib(2).
1
true.

?- generate_fib(3).
2
true.

?- generate_fib(4).
3
true.
```

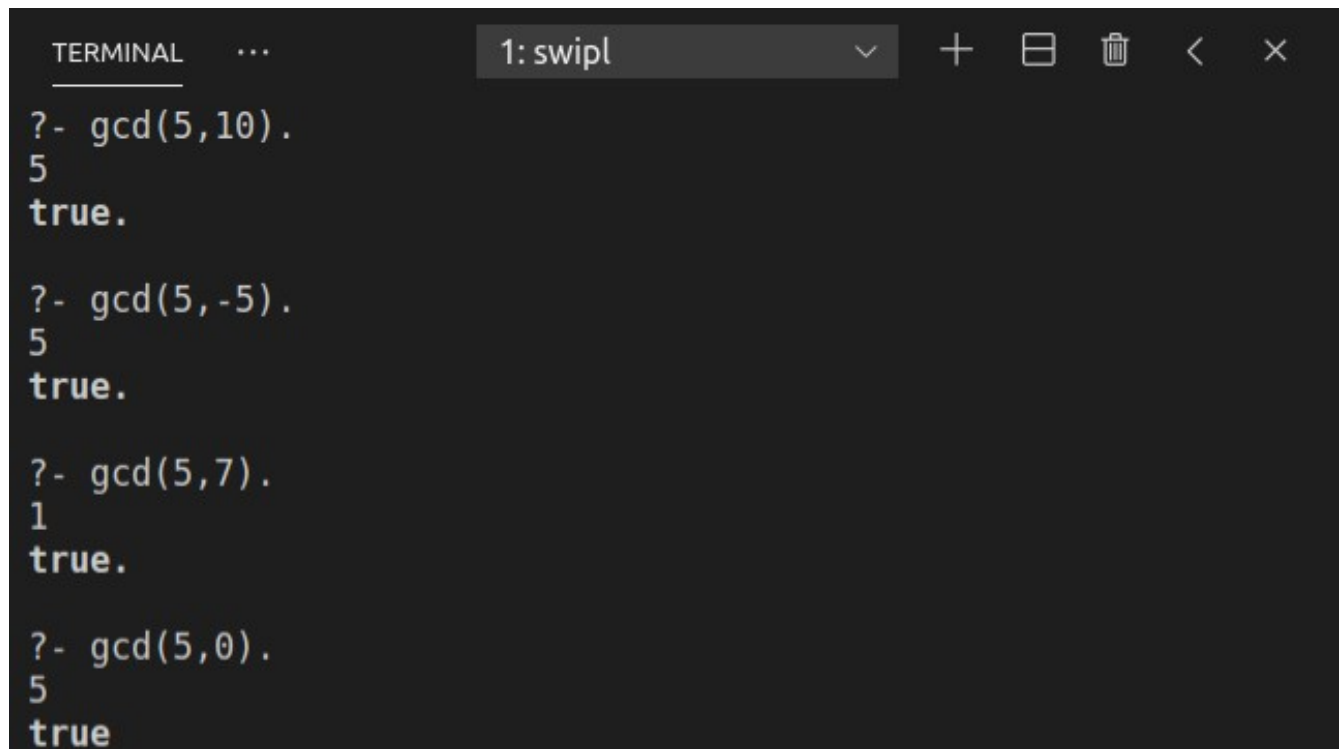
Figure 4: Solution 4 Image

Question 5

Write a prolog program to implement GCD of two numbers.

Solution 5

```
gcd(A,B):-A1 is abs(A), B1 is abs(B), gcd(A1, B1, N1), print(N1).
gcd(0,B,N):- N is B.
gcd(A,0,N):- N is A.
gcd(A,B,N):-
(   A = B
  -> N is A
  ;   (   A > B
        -> (   N1 is A-B,
              gcd(N1,B, G),
              N is G
            )
        ;   (   N1 is B - A,
              gcd(A, N1, G),
              N is G
            )
      )
).
```



The image shows a terminal window titled "1: swipl" with a dark background. It displays four Prolog queries and their results:

```
?- gcd(5,10).
5
true.

?- gcd(5,-5).
5
true.

?- gcd(5,7).
1
true.

?- gcd(5,0).
5
true
```

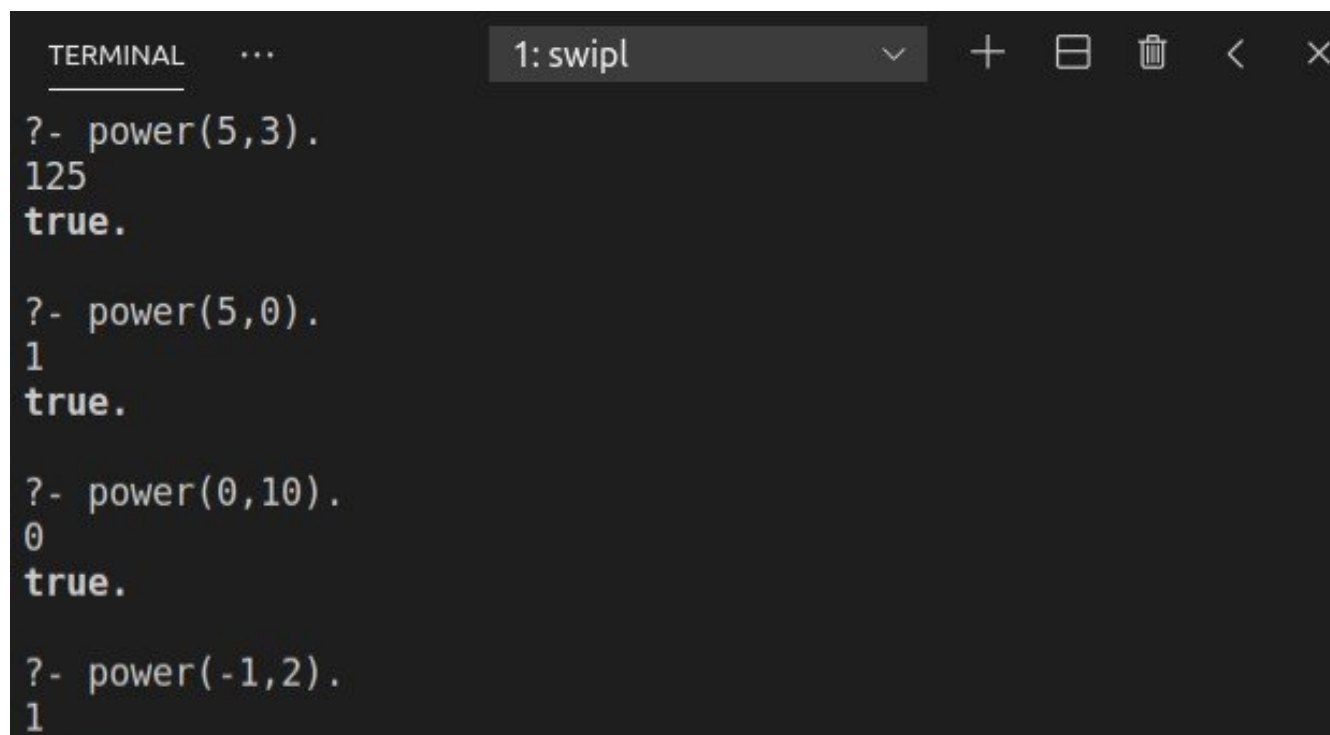
Figure 5: Solution 5 Image

Question 6

Write a prolog program to calculate the power of a given number N.

Solution 6

```
power(N, P):- power(N, P, A), print(A),!.
power(1, _, 1).
power(0, _, 0).
power(_, 0, 1).
power(N, P, A):-
(
    P > 0
    -> (
        P1 is P - 1,
        power(N, P1, A1),
        A is N*A1
    )
    ; (
        P1 is P + 1,
        power(N, P1, A1),
        A is 1/N*A1
    )
).
```



The screenshot shows a terminal window titled "1: swipl" with a dark background. The terminal displays the following Prolog queries and their results:

```
?- power(5,3).
125
true.

?- power(5,0).
1
true.

?- power(0,10).
0
true.

?- power(-1,2).
1
```

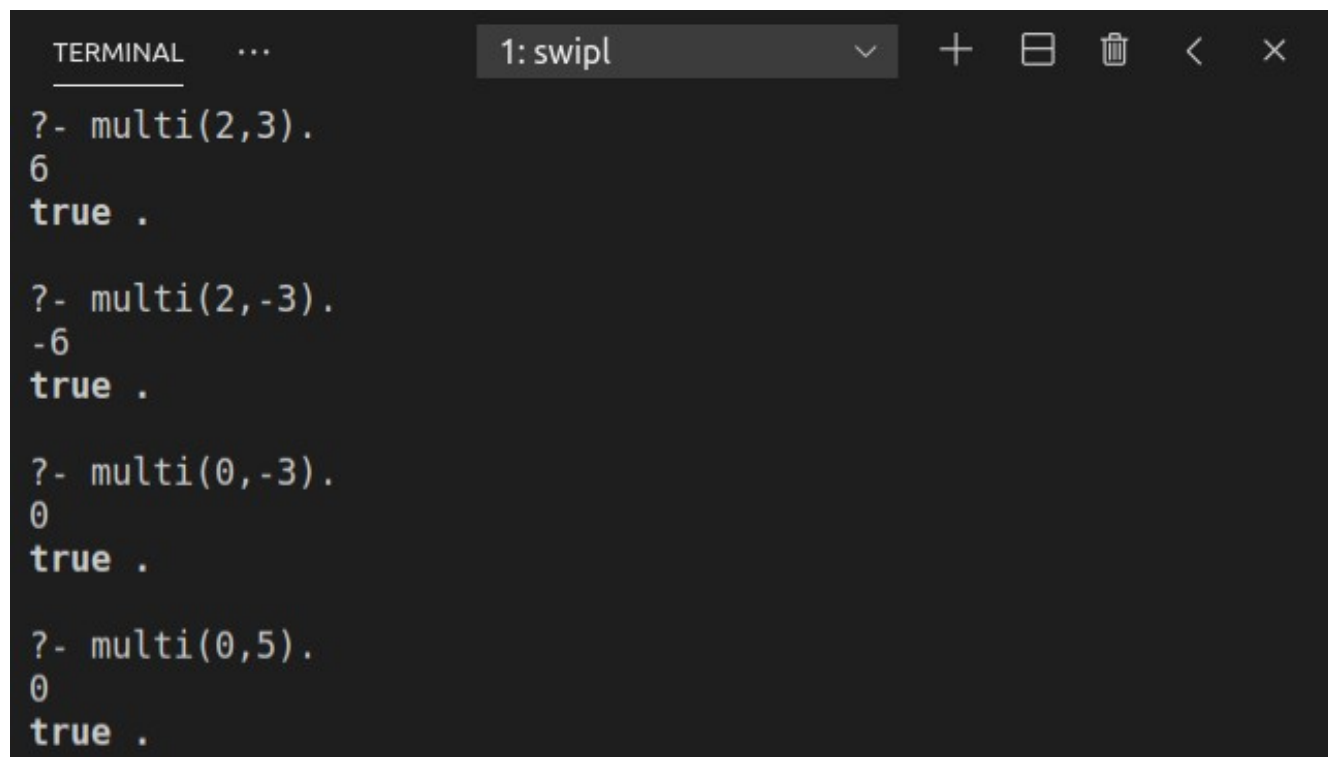
Figure 6: Solution 6 Image

Question 7

Write a prolog program to implement multiplication of two numbers.

Solution 7

```
multi(X,Y):-multi(X,Y,R), print(R).
multi(0,_,0).
multi(_,0,0).
multi(X,Y,R):-
(   Y > 0
    -> (
        Y1 is Y - 1,
        multi(X,Y1,R1),
        R is X + R1
    )
    ; (
        Y1 is Y + 1,
        multi(X,Y1,R1),
        R is -1*X + R1
    )
).
```



The screenshot shows a terminal window titled "1: swipl" with a toolbar containing icons for adding, saving, deleting, and navigating. The terminal displays the following Prolog queries and their results:

```
?- multi(2,3).
6
true .

?- multi(2,-3).
-6
true .

?- multi(0,-3).
0
true .

?- multi(0,5).
0
true .
```

Figure 7: Solution 7 Image

Question 8

Write a prolog program to implement towerofhanoi (N) where N represents the number of discs.

Solution 8

```
toh(N):-  
(  
    N < 0  
    -> print("Not defined")  
    ;   power(2, N, R),  
        R1 is R - 1,  
        write("No of steps: "),write(R1),nl,  
        toh(N, "a","b","c")  
).  
toh(1, A,_,C):-write("Move from "),write(A),write(" to "),write(C),nl.  
toh(N, A, B, C):-  
(  
    N1 is N-1,  
    toh(N1, A,C,B),  
    write("Move from "),write(A),write(" to "),write(C),nl,  
    toh(N1, B,A,C)  
).  
).
```

The image shows a terminal window titled "1: swipl" with a dark background. It displays the execution of a Prolog program for the Tower of Hanoi. The first query is `?- toh(2).`, which outputs "No of steps: 3" followed by three moves: "Move from a to b", "Move from a to c", and "Move from b to c", ending with "true .". The second query is `?- toh(3).`, which outputs "No of steps: 7" followed by seven moves: "Move from a to c", "Move from a to b", "Move from c to b", "Move from a to c", "Move from b to a", "Move from b to c", and "Move from a to c".

```
1: swipl  
?- toh(2).  
No of steps: 3  
Move from a to b  
Move from a to c  
Move from b to c  
true .  
  
?- toh(3).  
No of steps: 7  
Move from a to c  
Move from a to b  
Move from c to b  
Move from a to c  
Move from b to a  
Move from b to c  
Move from a to c
```

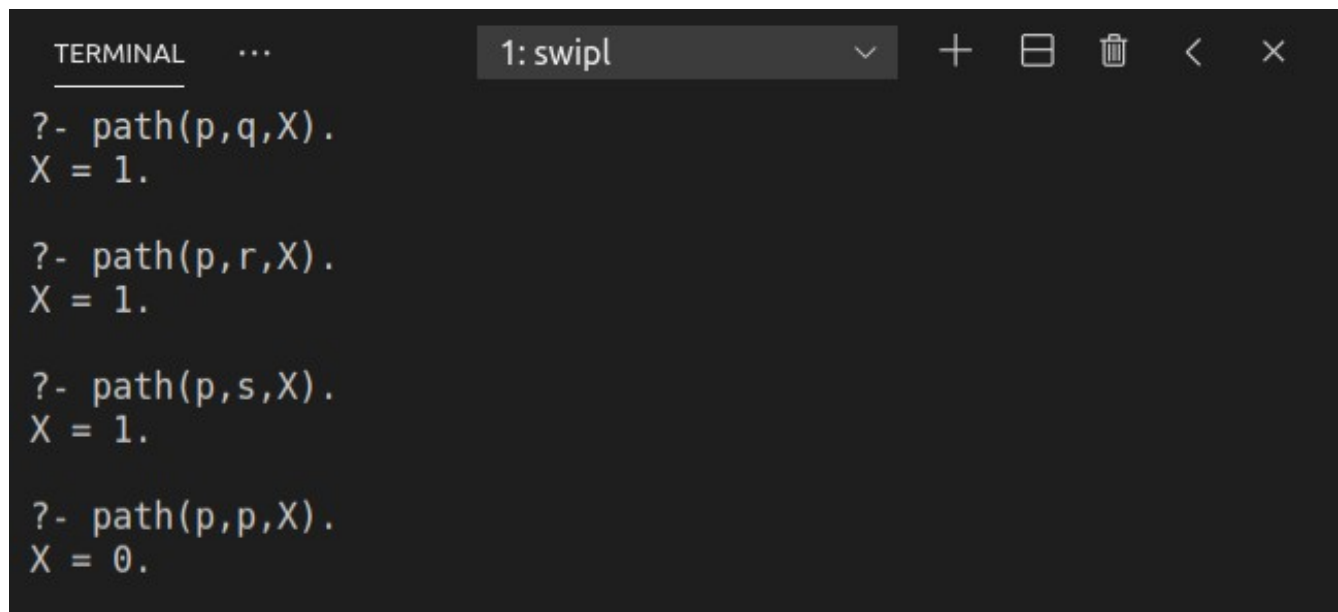
Figure 8: Solution 8 Image

Question 9

Consider a cyclic directed graph [edge (p, q), edge (q, r), edge (q, r), edge (q, s), edge (s,t)] where edge (A,B) is a predicate indicating directed edge in a graph from a node A to a node B. Write a program to check whether there is a route from one node to another node.

Solution 9

```
node(p). node(q). node(r). node(s). node(t).
edge(p,q). edge(q,r). edge(r,q). edge(q,s). edge(s,t).
path(X,Y,R):-
(
    node(X),
    node(Y),
    X \= Y
    -> (
        edge(X,Y)
        -> R is 1
        ; (
            edge(X,Z),
            Y \= Z,
            path(Z,Y, R2),
            R2 = 1
            -> R is 1
            ; R is 0
        )
    )
    ; R is 0
).
```



```
1: swipl
?- path(p,q,X).
X = 1.

?- path(p,r,X).
X = 1.

?- path(p,s,X).
X = 1.

?- path(p,p,X).
X = 0.
```

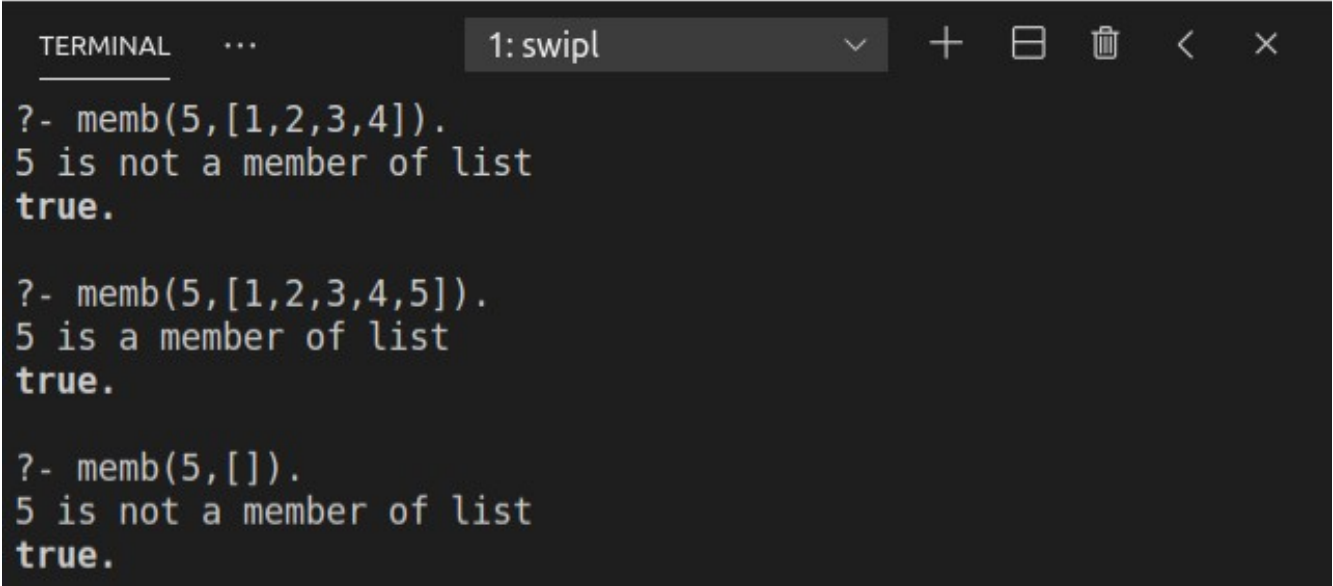
Figure 9: Solution 9 Image

Question 10

Write a prolog program to check whether X is a member of List L or not.

Solution 10

```
memb(A,L):-(  
    memb(A,L,R),  
    R = 1  
-> write(A), write(" is a member of list"),!  
    ; write(A), write(" is not a member of list"),!  
).  
memb(H, [H|_], 1).  
memb(X, [H|T], R):-  
(  
    X = H  
->R is 1  
    ; (  
        memb(X, T, R1),  
        R is R1  
    )  
).  
).
```



The image shows a terminal window with a dark background. The title bar at the top reads "1: swipl". The terminal contains three Prolog queries and their corresponding outputs:

```
?- memb(5,[1,2,3,4]).  
5 is not a member of list  
true.  
  
?- memb(5,[1,2,3,4,5]).  
5 is a member of list  
true.  
  
?- memb(5,[]).  
5 is not a member of list  
true.
```

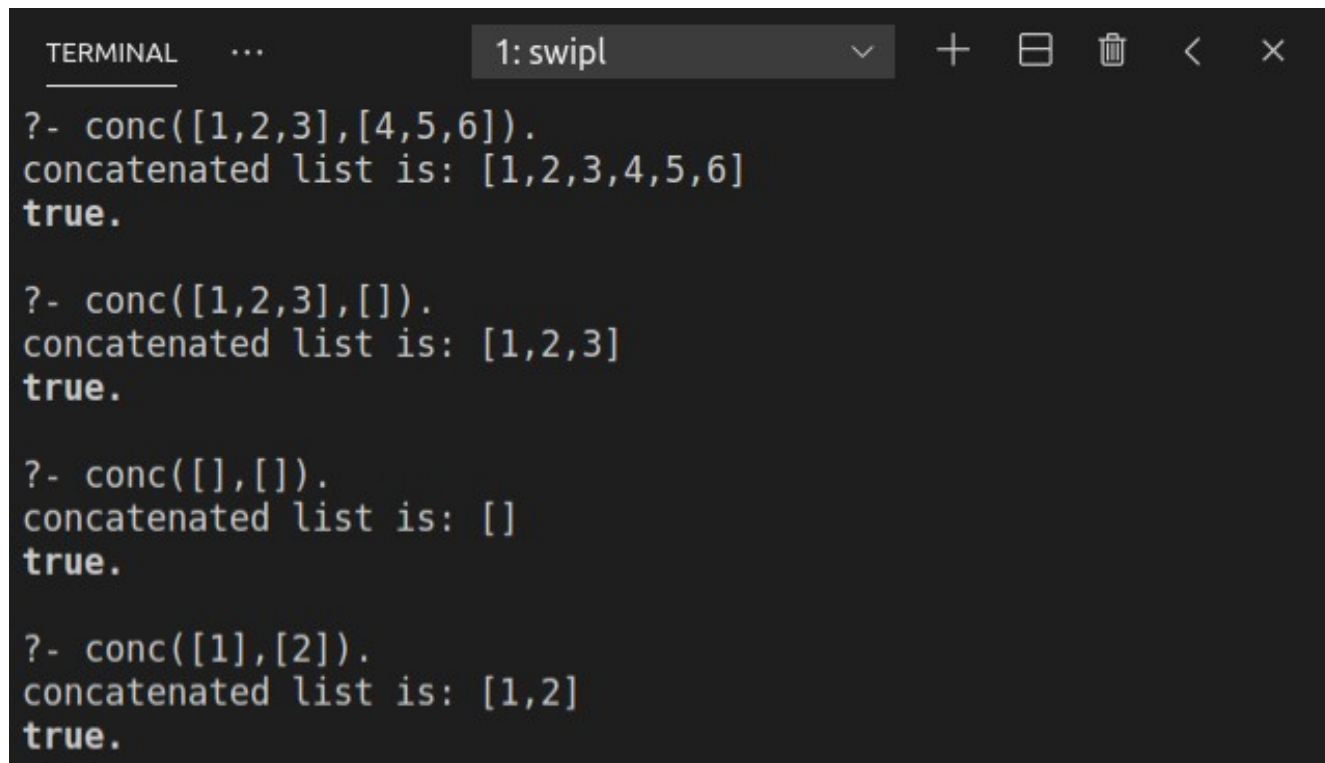
Figure 10: Solution 10 Image

Question 11

Write a prolog program to concatenate two lists to get the resultant list.

Solution 11

```
conc(A,B):-conc(A,B,R),write("concatenated list is: "),write(R),!.  
conc([], X, X).  
conc(X, [], X).  
conc([H1|T1], L2, [H1|T3]):- conc(T1, L2, T3).
```



```
1: swipl  
?- conc([1,2,3],[4,5,6]).  
concatenated list is: [1,2,3,4,5,6]  
true.  
  
?- conc([1,2,3],[]).  
concatenated list is: [1,2,3]  
true.  
  
?- conc([],[]).  
concatenated list is: []  
true.  
  
?- conc([1],[2]).  
concatenated list is: [1,2]  
true.
```

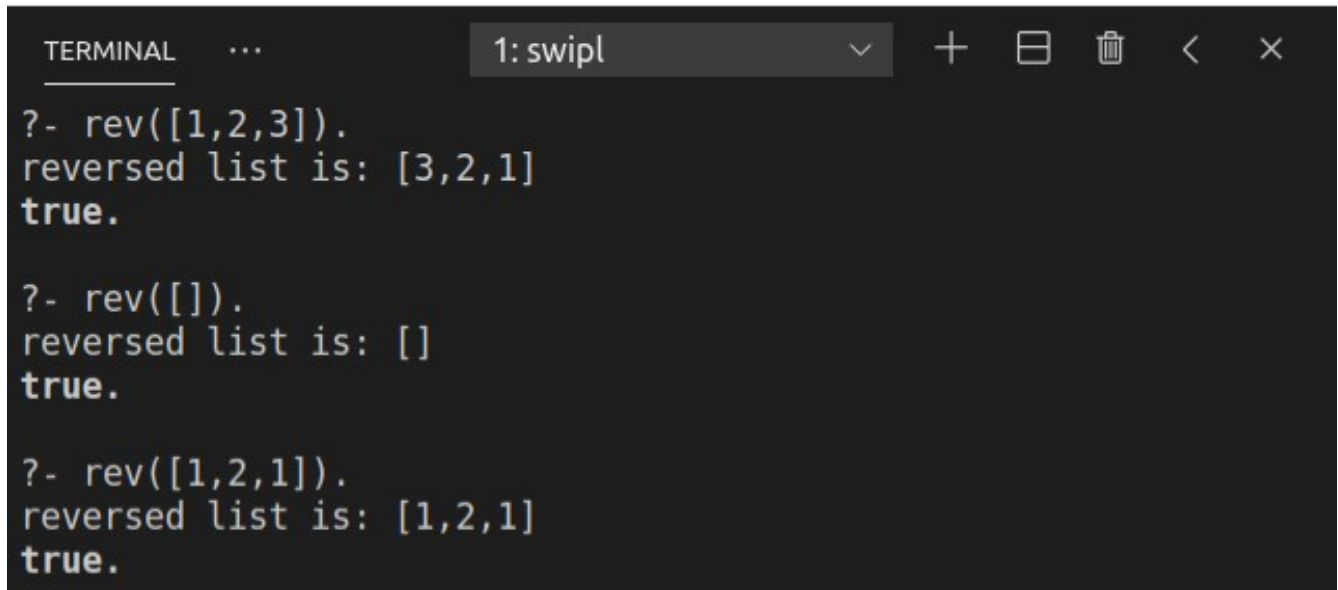
Figure 11: Solution 11 Image

Question 12

Write a prolog program to reverse a given list

Solution 12

```
rev(X):-rev(X,R),write("reversed list is: "),write(R).  
rev(X, R):-rev(X,[],R).  
rev([], X, X).  
rev([H1|T1], PREV, REV):-rev(T1, [H1|PREV], REV).
```



A terminal window titled "1: swipl" with a "TERMINAL" tab. It displays three Prolog queries and their results:

```
?- rev([1,2,3]).  
reversed list is: [3,2,1]  
true.  
  
?- rev([]).  
reversed list is: []  
true.  
  
?- rev([1,2,1]).  
reversed list is: [1,2,1]  
true.
```

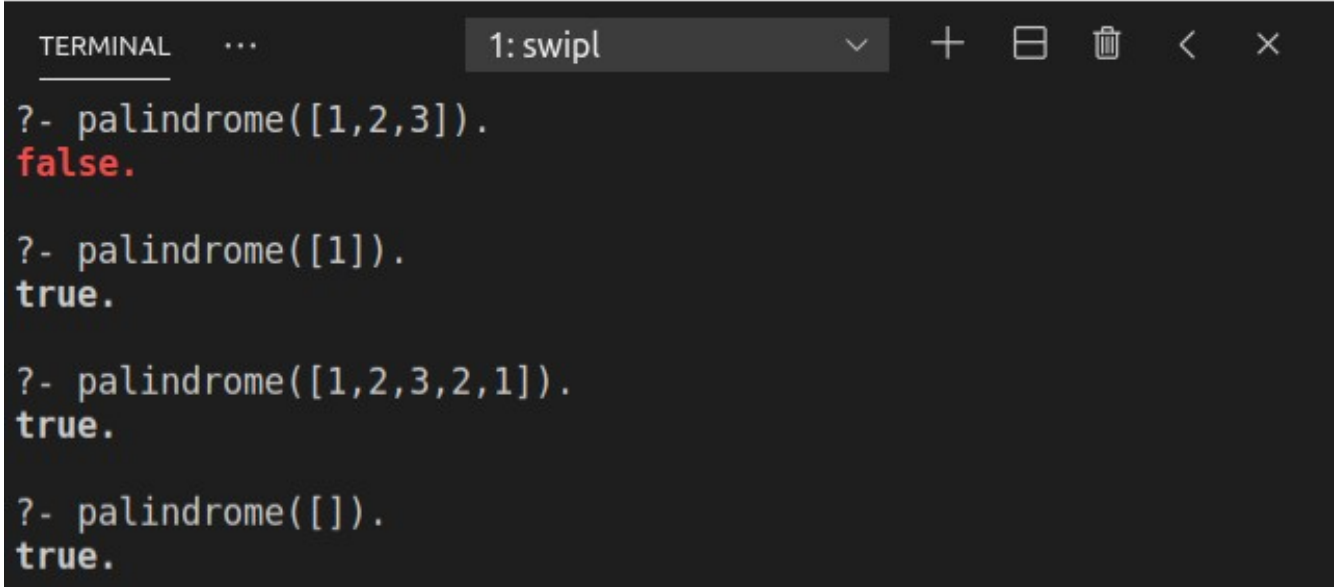
Figure 12: Solution 12 Image

Question 13

Write a prolog program to check whether a list L is a palindrome or not

Solution 13

```
rev(X, R):-rev(X,[],R).  
rev([], X, X).  
rev([H1|T1], PREV, REV):-rev(T1, [H1|PREV], REV).  
palindrome(A):-rev(A,R), A = R.
```



The image shows a terminal window with a dark background. The title bar at the top reads "1: swipl". The terminal content shows four Prolog queries and their results:

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

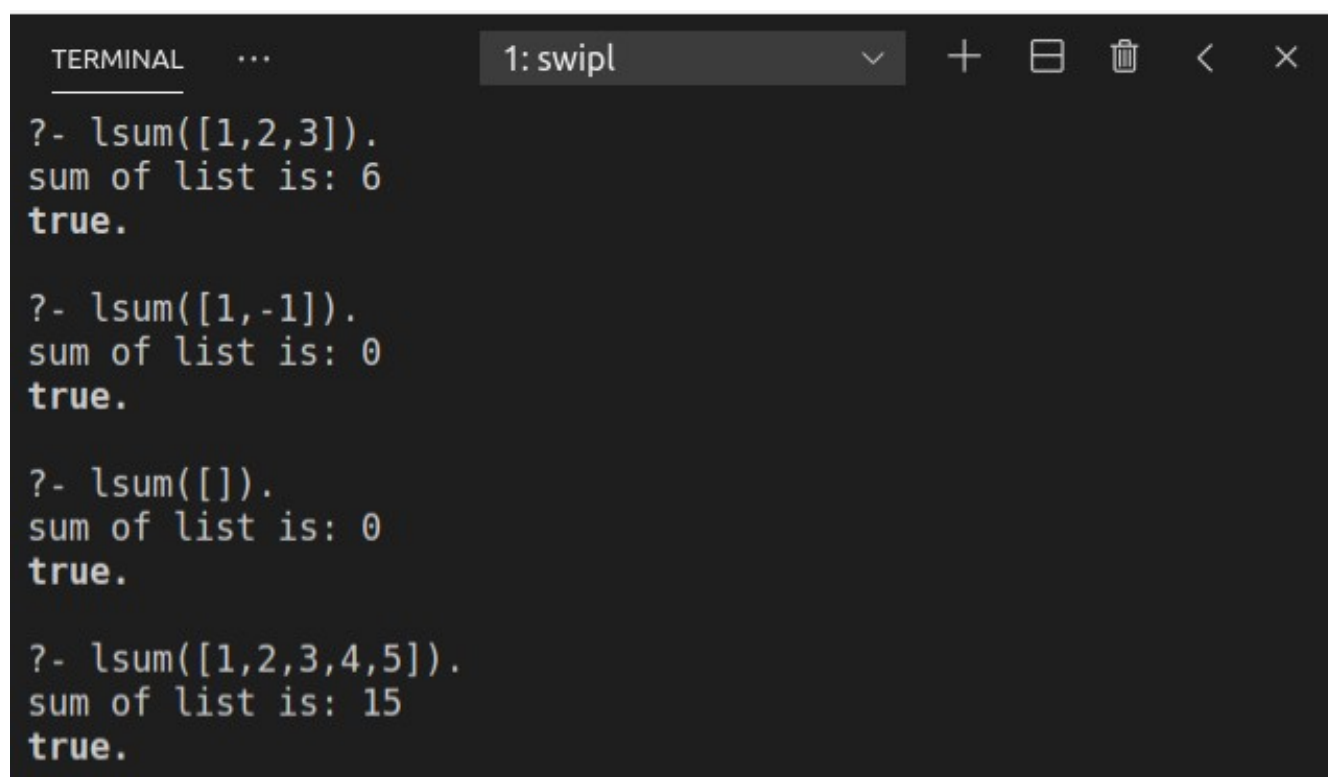
Figure 13: Solution 13 Image

Question 14

Write a prolog program to find the sum of a given list L.

Solution 14

```
lsum(L):-lsum(L,S),write("sum of list is: "),write(S).  
lsum([], 0).  
lsum([H|T], R):-  
(  
    lsum(T, R1),  
    R is H + R1  
).  
).
```



The image shows a terminal window with a dark background. The title bar at the top includes the word "TERMINAL", a menu icon, and a tab labeled "1: swipl". The terminal displays four Prolog queries and their corresponding outputs:

```
?- lsum([1,2,3]).  
sum of list is: 6  
true.  
  
?- lsum([1,-1]).  
sum of list is: 0  
true.  
  
?- lsum([]).  
sum of list is: 0  
true.  
  
?- lsum([1,2,3,4,5]).  
sum of list is: 15  
true.
```

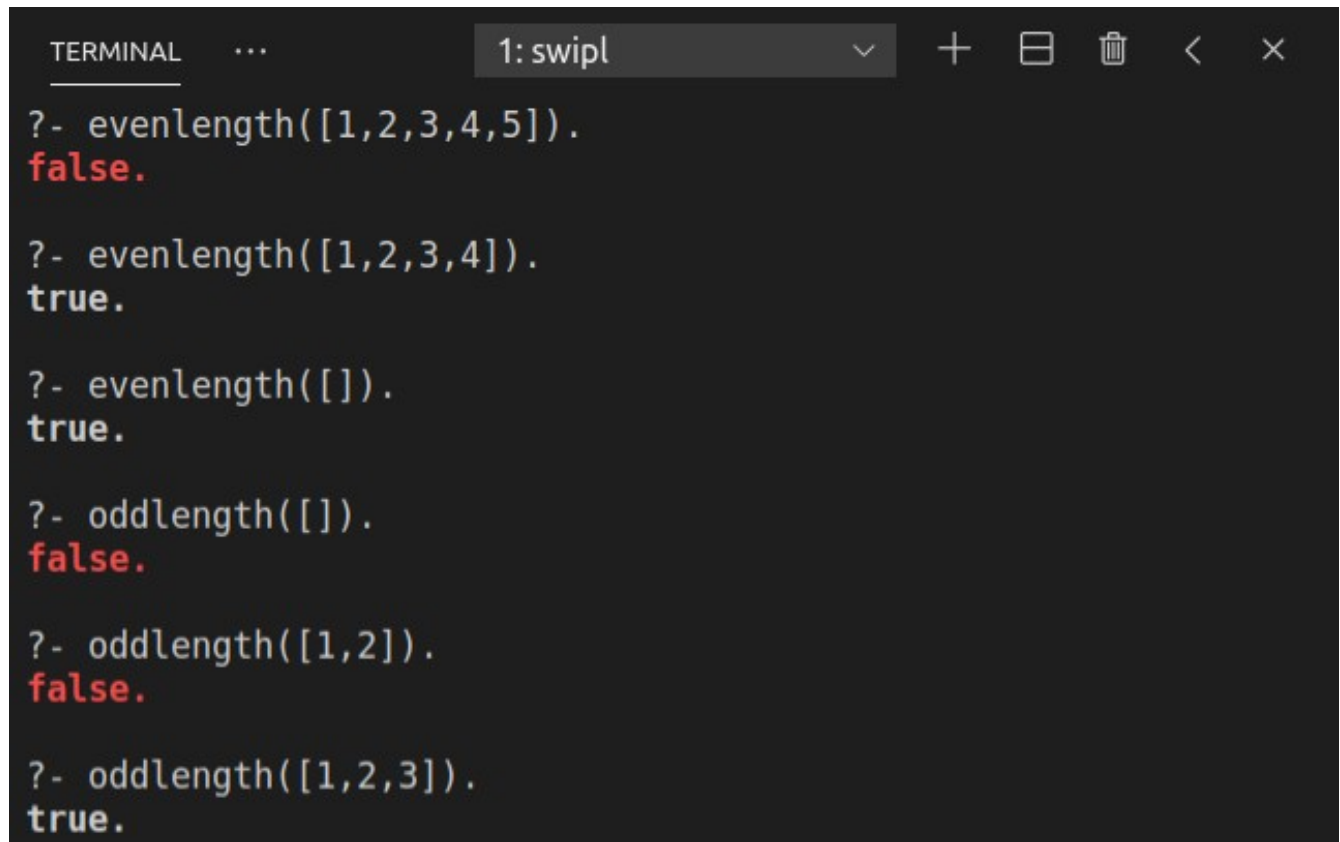
Figure 14: Solution 14 Image

Question 15

Write a prolog program to write two methods, so that they return true if their argument is a list of even or odd length respectively.

Solution 15

```
evenlength(L):-  
(  
    len(L, R1),  
    0 is mod(R1,2)  
).  
oddlength(L):-  
(  
    len(L, R1),  
    1 is mod(R1,2)  
).
```



```
1: swipl  
?- evenlength([1,2,3,4,5]).  
false.  
  
?- evenlength([1,2,3,4]).  
true.  
  
?- evenlength([]).  
true.  
  
?- oddlength([]).  
false.  
  
?- oddlength([1,2]).  
false.  
  
?- oddlength([1,2,3]).  
true.
```

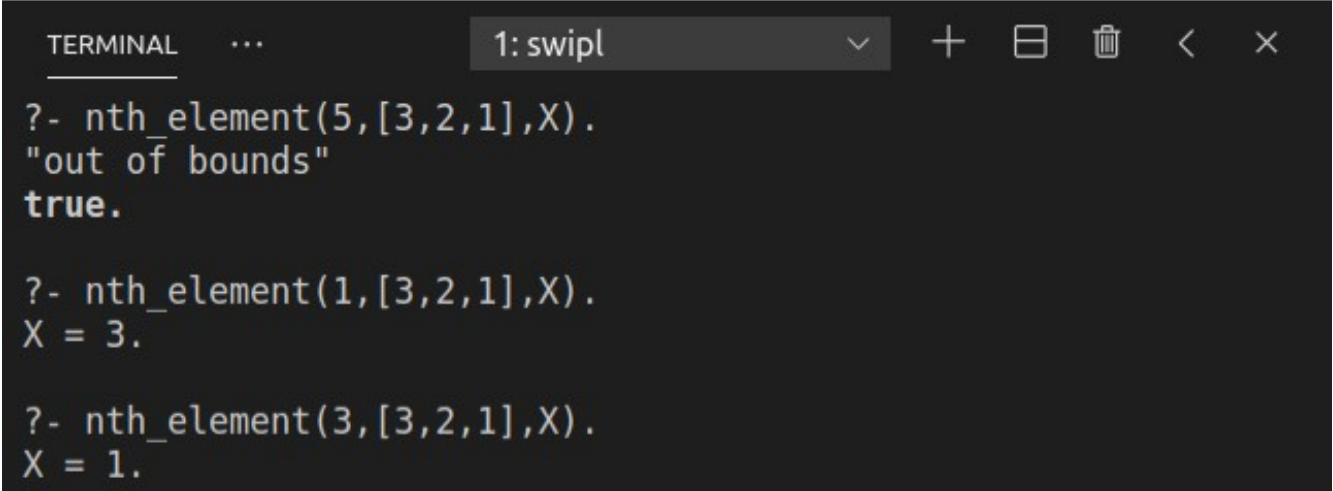
Figure 15: Solution 15 Image

Question 16

Write a prolog program to return element at position N in a list L.

Solution 16

```
nth_element(N, L):-nth_element(N, L, X),write("element at pos "),write(N),write(" is: "),write(X),!.
nth_element(_, [], _):-print("out of bounds"),!.
nth_element(N, [H|T], X):-
(
    N > 0
    -> (
        N = 1
        ->X is H
        ; (
            N1 is N - 1,
            nth_element(N1, T, X),!
        )
    )
; print("Invalid index")
).
```



```
1: swipl
?- nth_element(5,[3,2,1],X).
"out of bounds"
true.

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

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

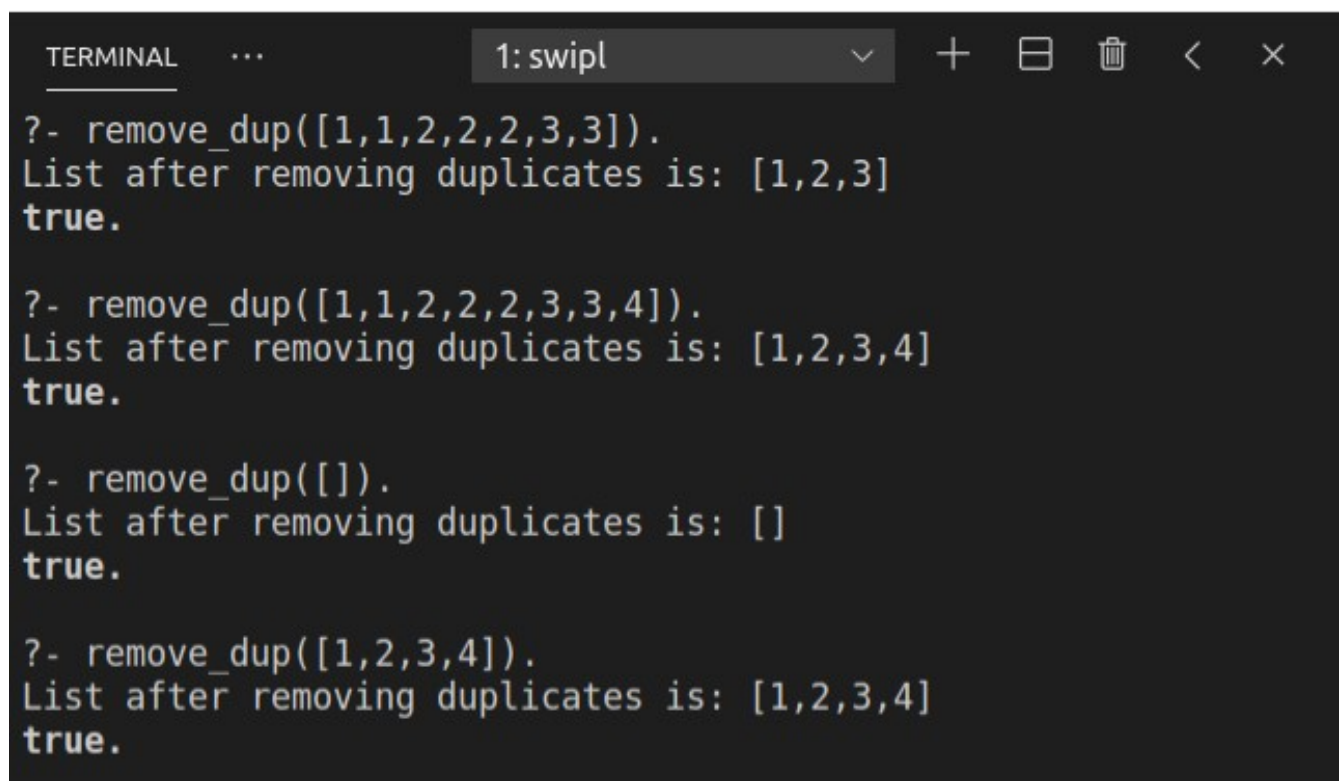
Figure 16: Solution 16 Image

Question 17

Write a prolog program to remove duplicates from a given list L.

Solution 17

```
remove_dup(L):-remove_dup(L, R),write("List after removing duplicates is: "),write(R),!.
remove_dup([], []).
remove_dup([H|T], [H|R]):-
(
    delete_all(H, T, R1),
    remove_dup(R1, R)
).
```



The screenshot shows a terminal window titled "1: swipl" with a "TERMINAL" tab. It displays four Prolog queries and their corresponding outputs. Each query is followed by the message "List after removing duplicates is:" and the resulting list, followed by "true.".

```
1: swipl
?- remove_dup([1,1,2,2,2,3,3]).
List after removing duplicates is: [1,2,3]
true.

?- remove_dup([1,1,2,2,2,3,3,4]).
List after removing duplicates is: [1,2,3,4]
true.

?- remove_dup([]).
List after removing duplicates is: []
true.

?- remove_dup([1,2,3,4]).
List after removing duplicates is: [1,2,3,4]
true.
```

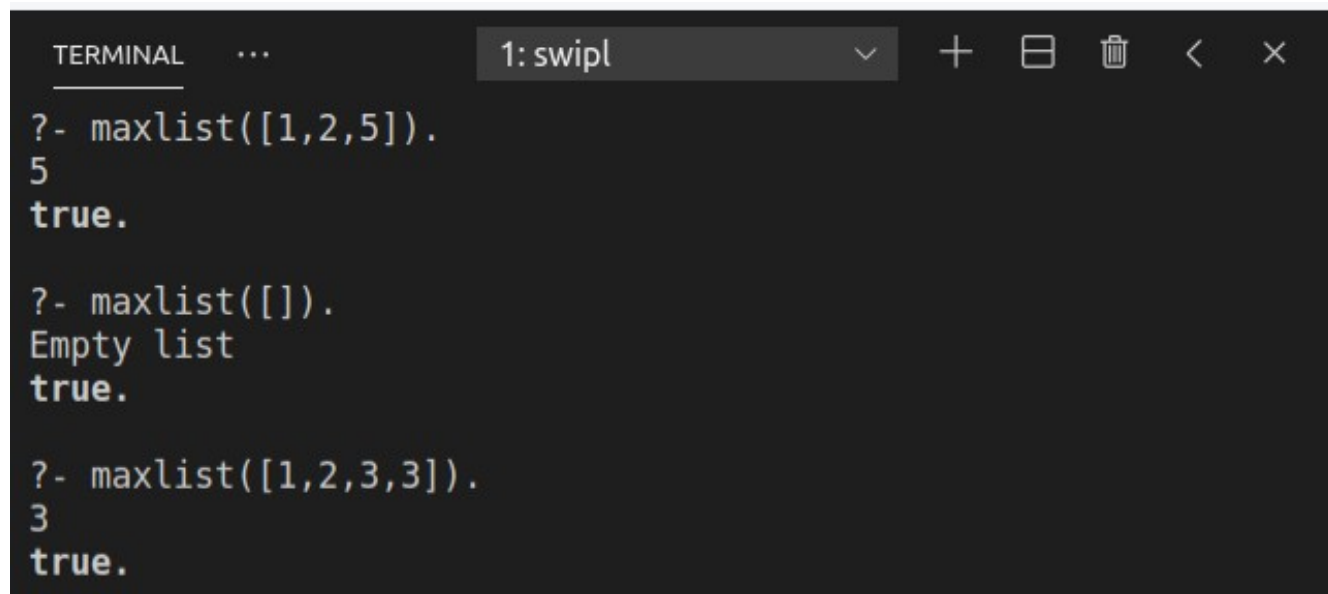
Figure 17: Solution 17 Image

Question 18

Write a prolog program which returns the maximum number in a list.

Solution 18

```
maxlist([]):-write("Empty list"),!.
maxlist([H|T]):-maxlist(T, H).
maxlist([H|[]], M):-
(
    M > H
    -> print(M),!
    ; print(H),!
).
maxlist([H|T], M):-
(
    M > H
    ->maxlist(T, M)
    ; maxlist(T, H)
).
```



A terminal window titled "1: swipl" with standard window controls. It displays three Prolog queries and their results:

```
?- maxlist([1,2,5]).
5
true.

?- maxlist([]).
Empty list
true.

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

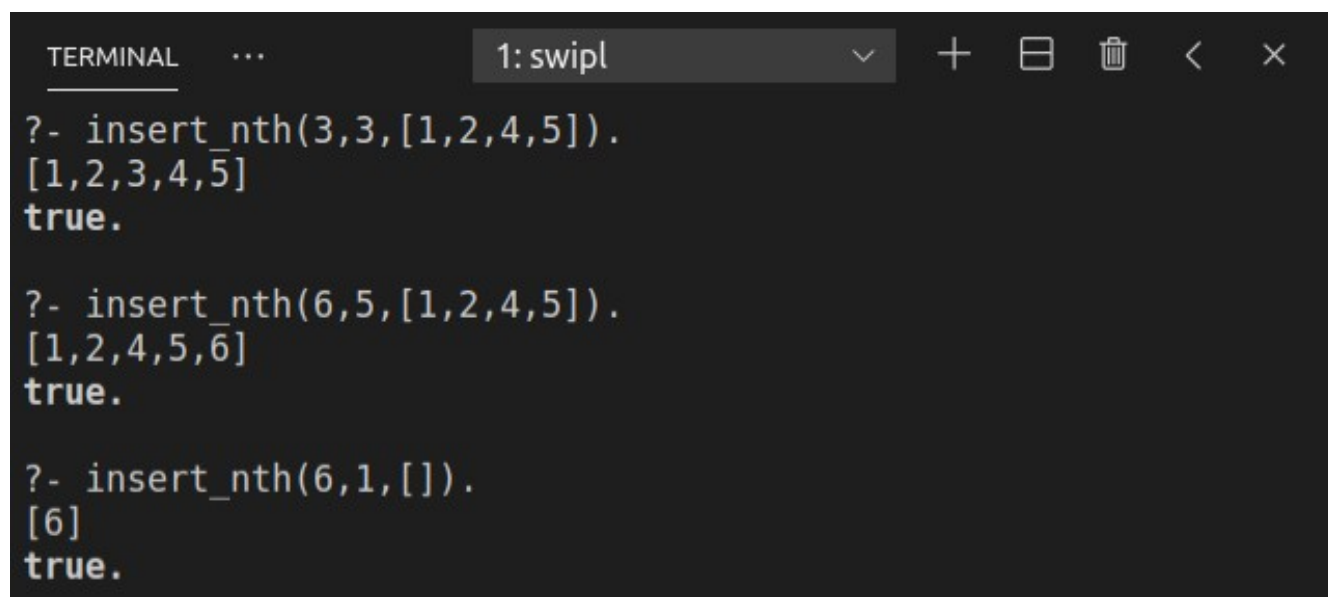
Figure 18: Solution 18 Image

Question 19

Write a prolog program to inserts an element I into Nth position of list L.

Solution 19

```
insert_nth(X, P, L):-insert_nth(X, P, L, R),print(R),!.
insert_nth(X, 1, Y, [X|Y]).
insert_nth(X, P, [H|T], [H|T1]):-
(
    P1 is P - 1,
    insert_nth(X, P1, T, T1)
).
```



A terminal window titled "1: swipl" showing three Prolog queries and their results. The queries are: 1. insert_nth(3,3,[1,2,4,5]). which returns [1,2,3,4,5] and true. 2. insert_nth(6,5,[1,2,4,5]). which returns [1,2,4,5,6] and true. 3. insert_nth(6,1,[]). which returns [6] and true. The terminal window has a dark background and standard window controls at the top.

```
1: swipl
?- insert_nth(3,3,[1,2,4,5]).
[1,2,3,4,5]
true.

?- insert_nth(6,5,[1,2,4,5]).
[1,2,4,5,6]
true.

?- insert_nth(6,1,[]).
[6]
true.
```

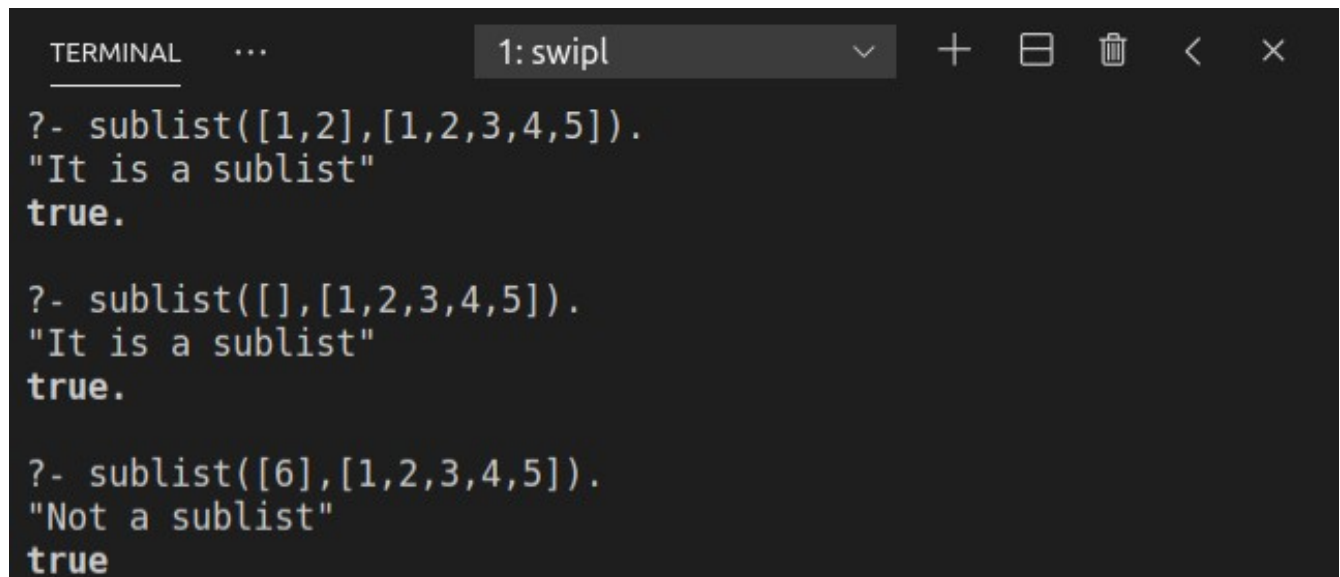
Figure 19: Solution 19 Image

Question 20

Write a prolog program to check whether the list S is the sublist of list L or not.

Solution 20

```
sublist([],[]):-print("It is a sublist").
sublist([],[_|_]):-print("It is a sublist").
sublist([_|_],[]):-print("Not a sublist").
sublist([H|T],[H1|T1]):-
(
    H = H1
    -> sublist(T, T1)
    ; sublist([H|T], T1)
).
```



```
1: swipl
?- sublist([1,2],[1,2,3,4,5]).
"It is a sublist"
true.

?- sublist([], [1,2,3,4,5]).
"It is a sublist"
true.

?- sublist([6],[1,2,3,4,5]).
"Not a sublist"
true
```

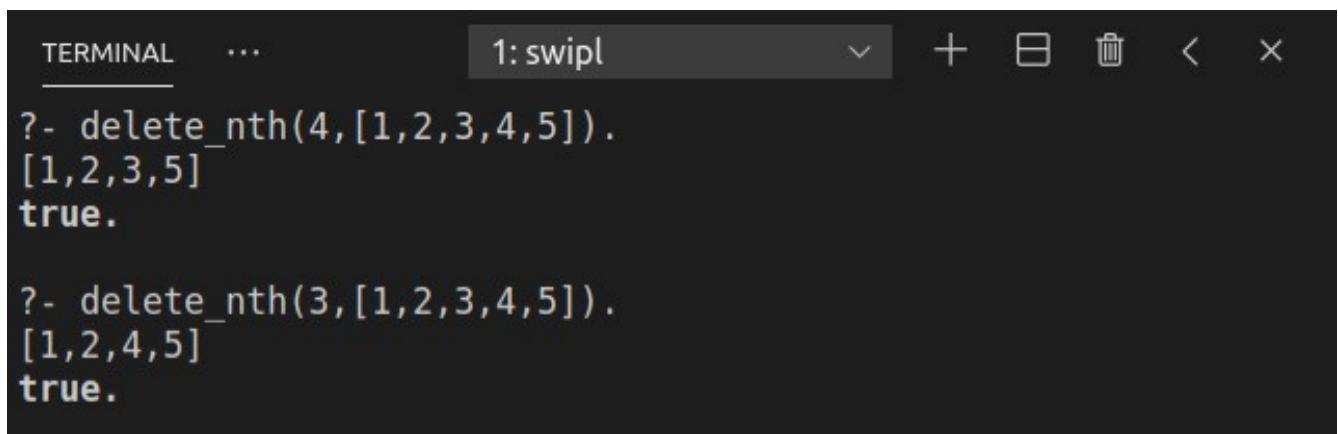
Figure 20: Solution 20 Image

Question 21

Write a prolog program that removes the element on Nth position from a list L.

Solution 21

```
delete_nth(P, L):-delete_nth(P, L,R), print(R),!.  
delete_nth(1, [_|T], T).  
delete_nth(P, [H|T], [H|T1]):-  
(  
    P1 is P - 1,  
    delete_nth(P1, T, T1)  
).  
.
```

A terminal window titled '1: swipl' with a dark background and light-colored text. It shows two Prolog queries and their results. The first query is '?- delete_nth(4,[1,2,3,4,5]).' which returns '[1,2,3,5]' and 'true.'. The second query is '?- delete_nth(3,[1,2,3,4,5]).' which returns '[1,2,4,5]' and 'true.'. The terminal window has standard icons for window management at the top right.

```
1: swipl  
?- delete_nth(4,[1,2,3,4,5]).  
[1,2,3,5]  
true.  
  
?- delete_nth(3,[1,2,3,4,5]).  
[1,2,4,5]  
true.
```

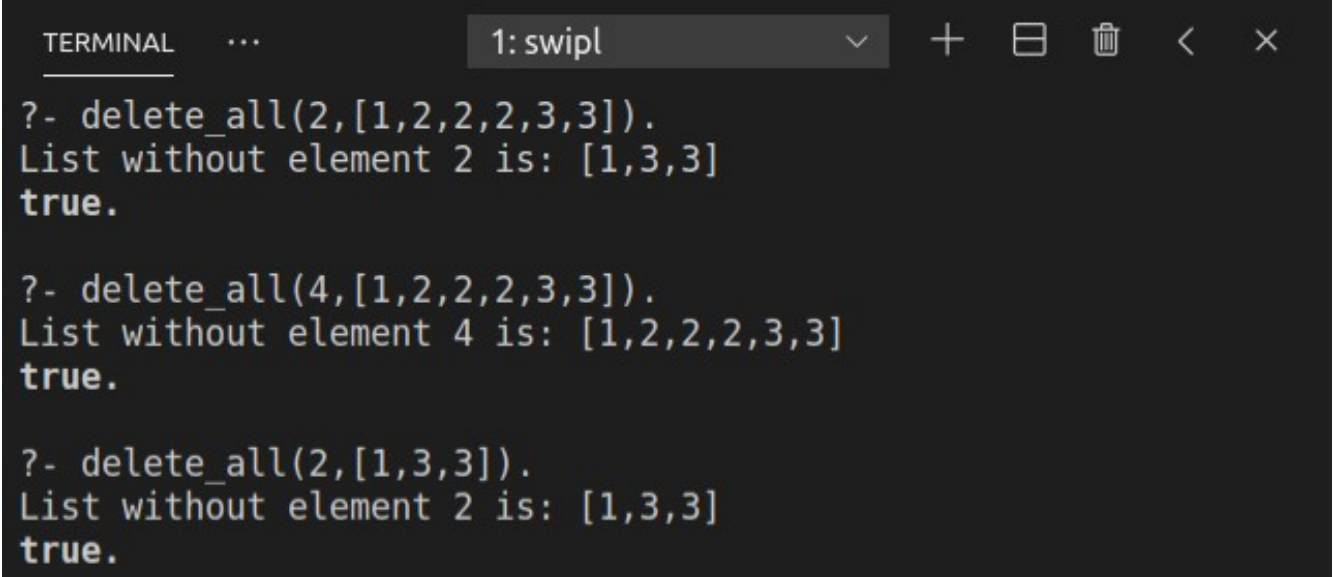
Figure 21: Solution 21 Image

Question 22

Write a prolog program to delete X where X denotes the element whose all occurrences has to be deleted from list L

Solution 22

```
delete_all(X, L):-delete_all(X,L,R),write("List without element "), write(X),write(" is: "),write(R),!.
delete_all(_, [], []).
delete_all(X, [X|T], L):-delete_all(X, T, L).
delete_all(X, [H|T], [H|T1]):-delete_all(X, T, T1).
```



```
1: swipl
?- delete_all(2,[1,2,2,2,3,3]).
List without element 2 is: [1,3,3]
true.

?- delete_all(4,[1,2,2,2,3,3]).
List without element 4 is: [1,2,2,2,3,3]
true.

?- delete_all(2,[1,3,3]).
List without element 2 is: [1,3,3]
true.
```

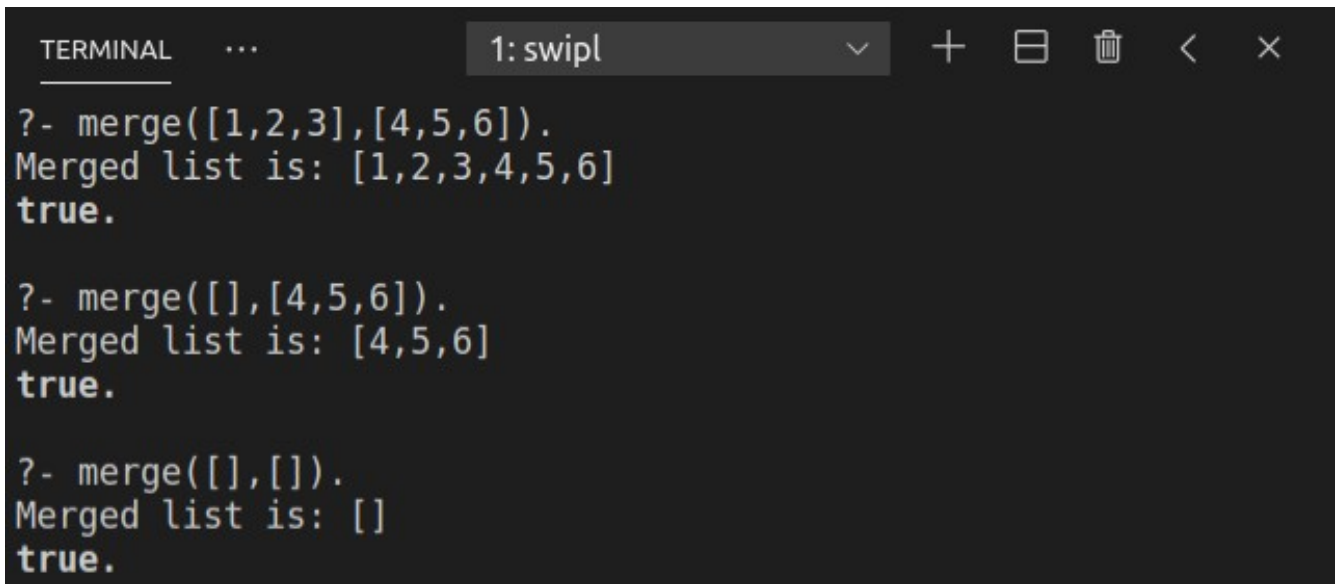
Figure 22: Solution 22 Image

Question 23

Write a prolog program to merge two lists.

Solution 23

```
merge(X, Y):-merge(X, Y, R), write("Merged list is: "),write(R),!.
merge([], X, X).
merge(X, [], X).
merge([H1|T1], [H2|T2], [X|R]):-
(
    H1 < H2
    -> X is H1,
        merge(T1, [H2|T2], R)
    ; X is H2,
        merge([H1|T1], T2, R)
).
```

A terminal window titled "1: swipl" with standard window controls. It displays three Prolog queries and their outputs. The first query merges [1,2,3] and [4,5,6] into [1,2,3,4,5,6]. The second merges an empty list with [4,5,6] into [4,5,6]. The third merges two empty lists into an empty list. Each output line is preceded by "Merged list is: " and ends with "true.".

```
TERMINAL  ...  1: swipl  +  -  x
?- merge([1,2,3],[4,5,6]).
Merged list is: [1,2,3,4,5,6]
true.

?- merge([], [4,5,6]).
Merged list is: [4,5,6]
true.

?- merge([], []).
Merged list is: []
true.
```

Figure 23: Solution 23 Image