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

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)
).
```

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)
).
```

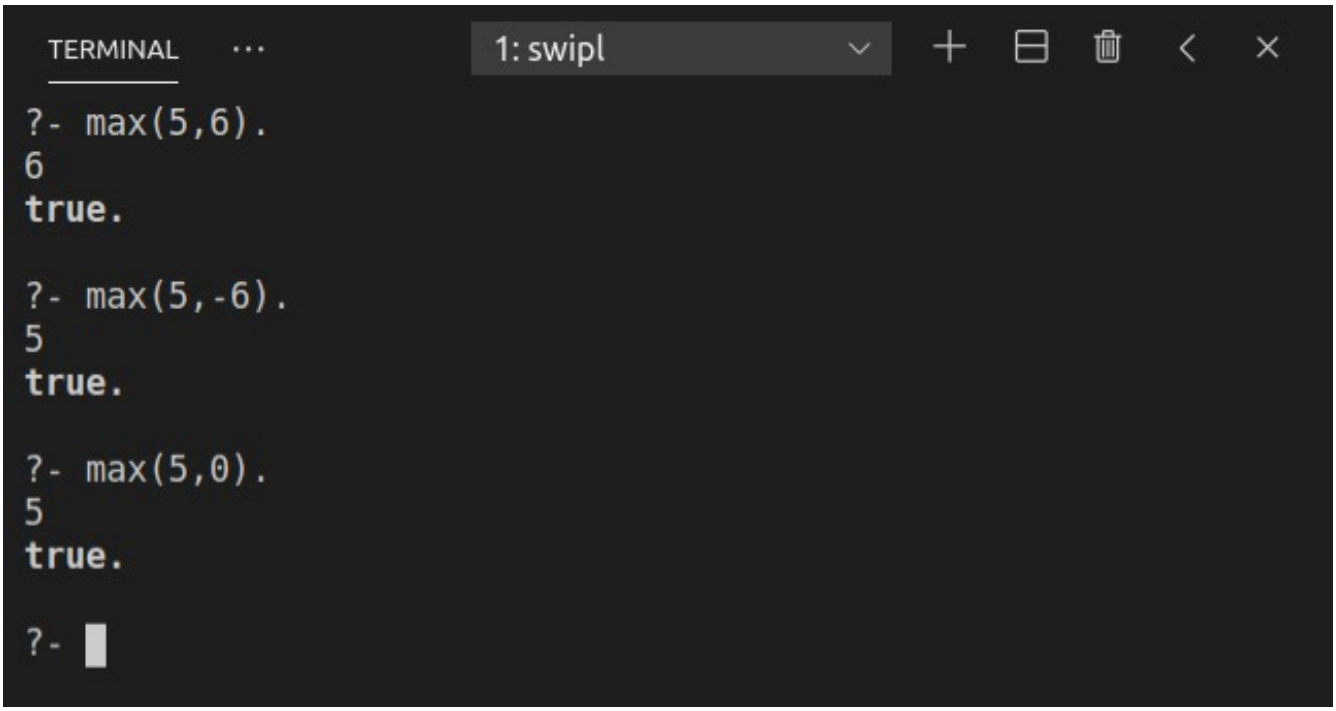

Figure 2: Solution 2 Image

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"),!
).
```

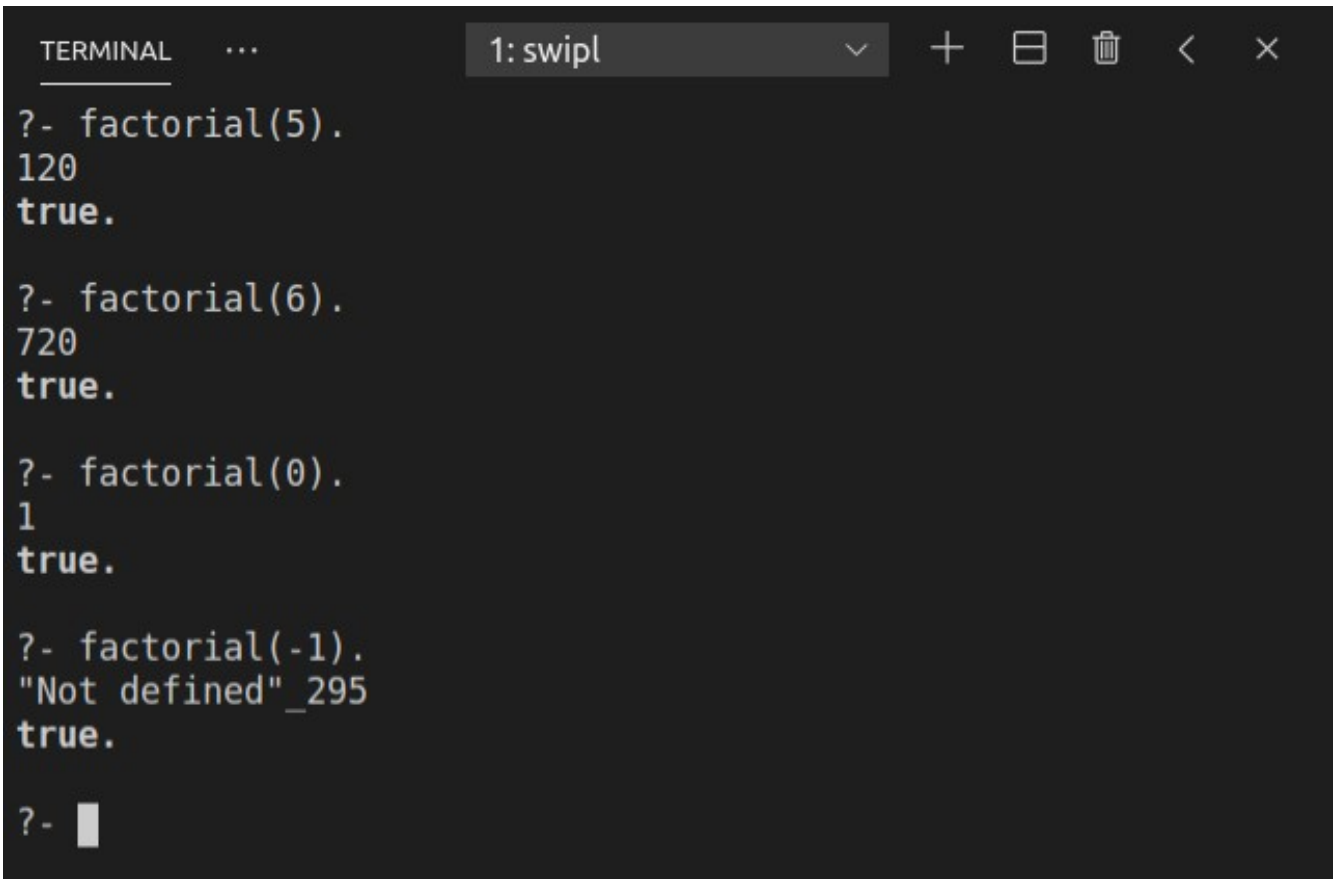

Figure 3: Solution 3 Image

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"),!
).
```

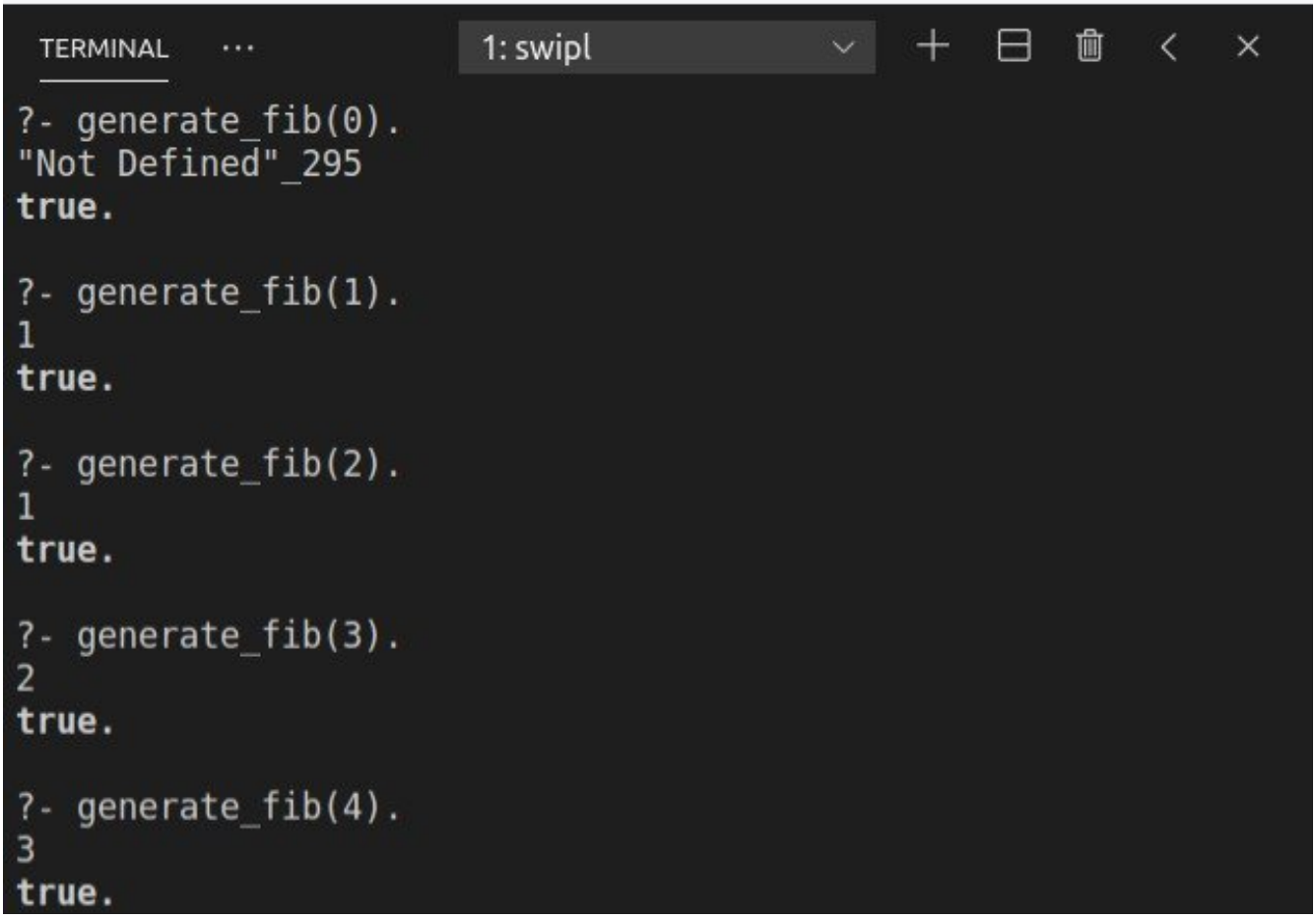

Figure 4: Solution 4 Image

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
                )
        )
).
```

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
        )

).
```

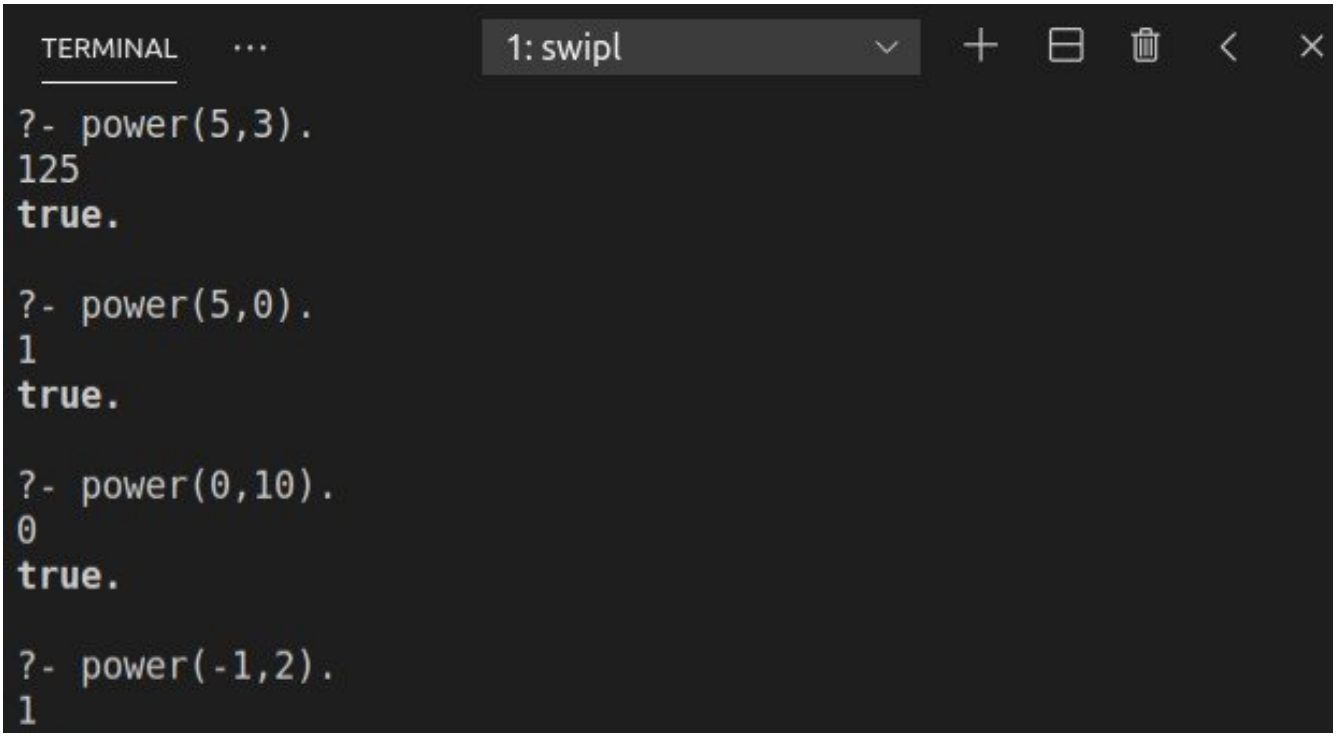

Figure 6: Solution 6 Image

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
        )

).
```

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)
).
```

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
).
```

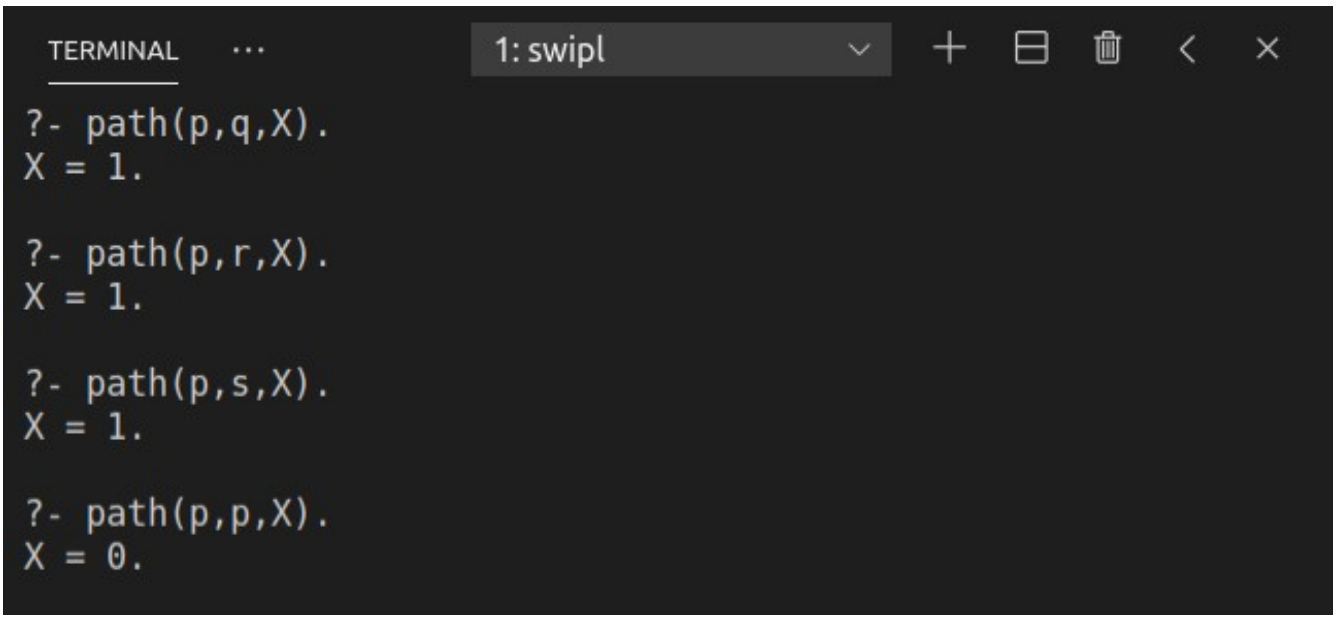

Figure 9: Solution 9 Image

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
        )
).
```

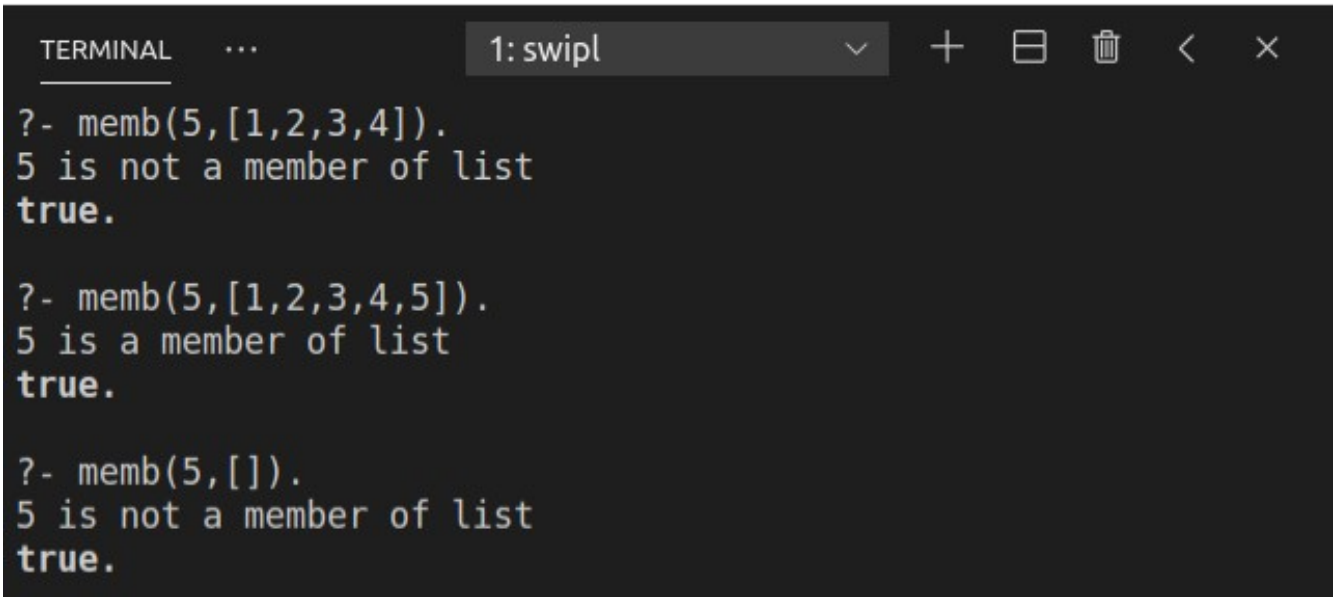

Figure 10: Solution 10 Image

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).
```

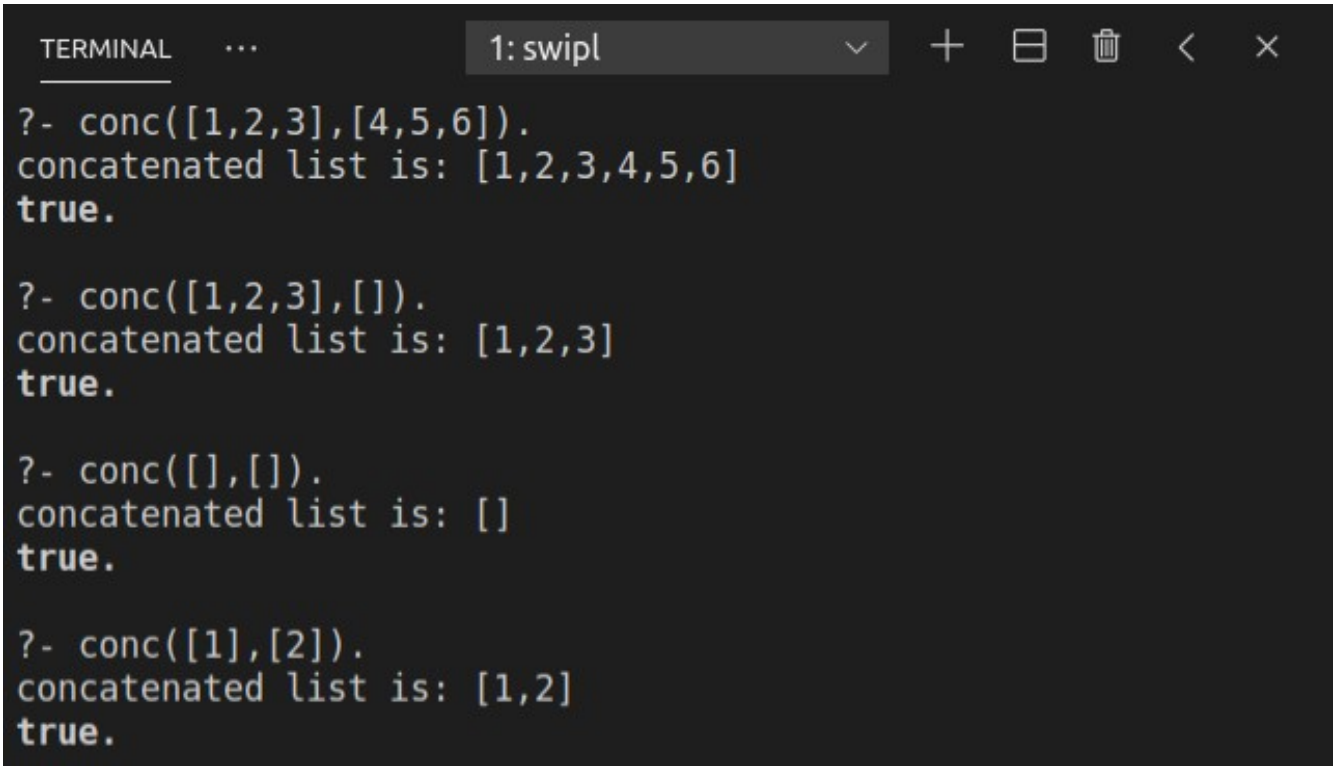

Figure 11: Solution 11 Image

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).
```

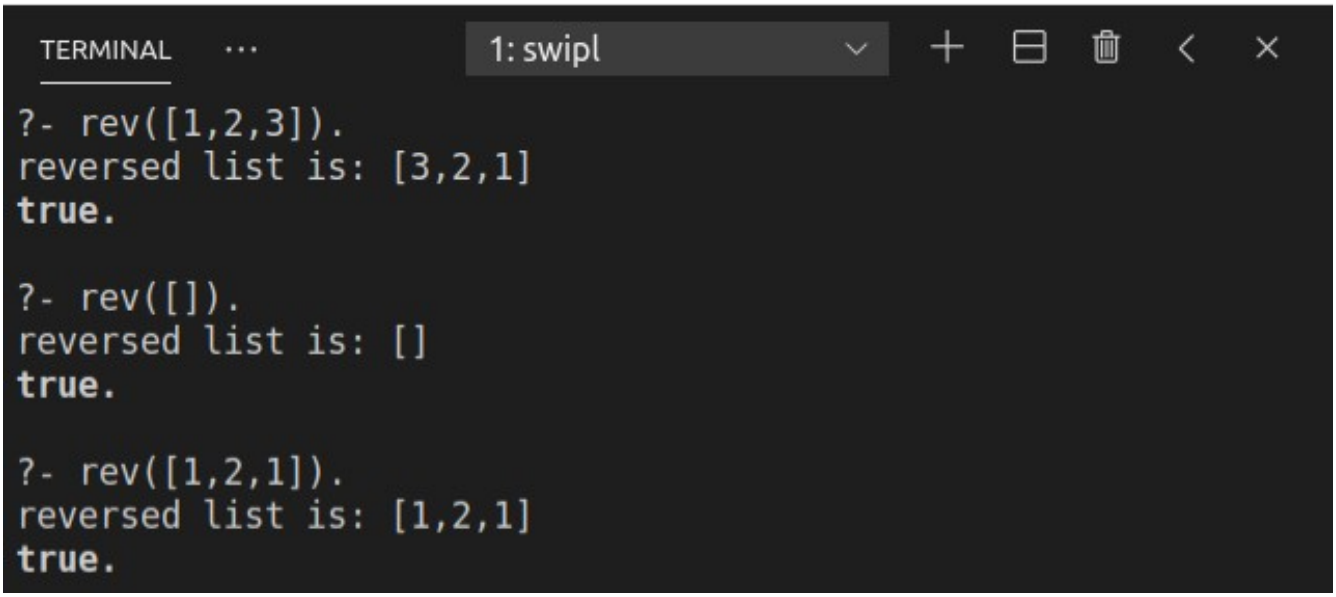

Figure 12: Solution 12 Image

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.
```

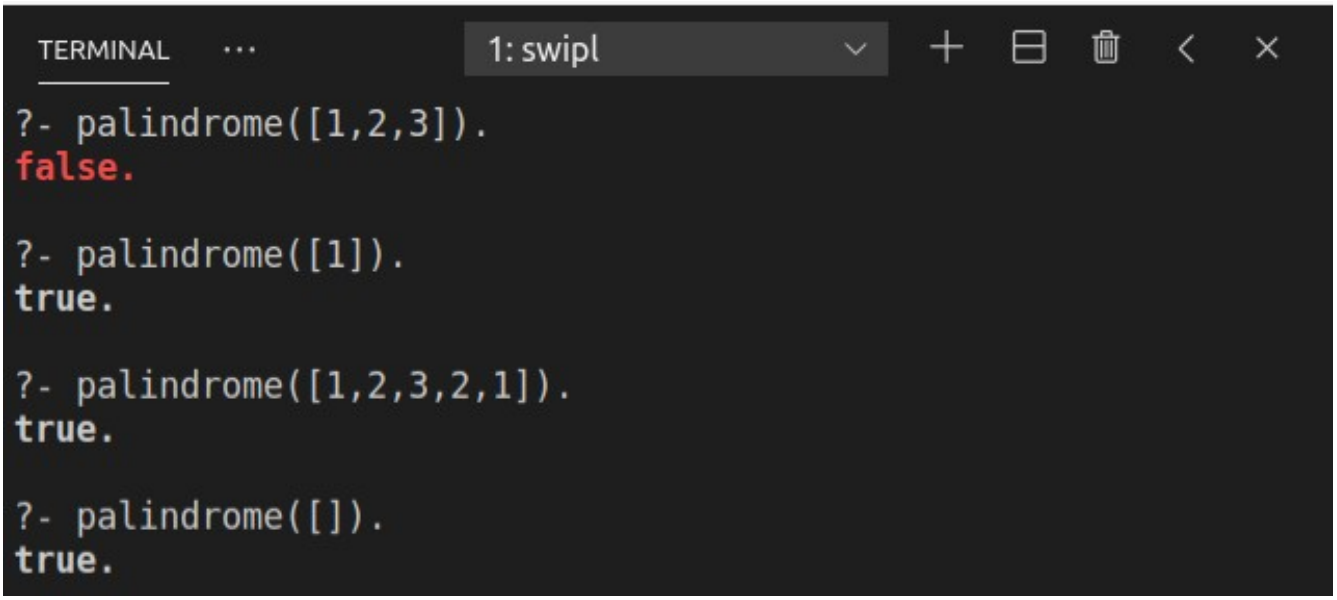

Figure 13: Solution 13 Image

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
).
```

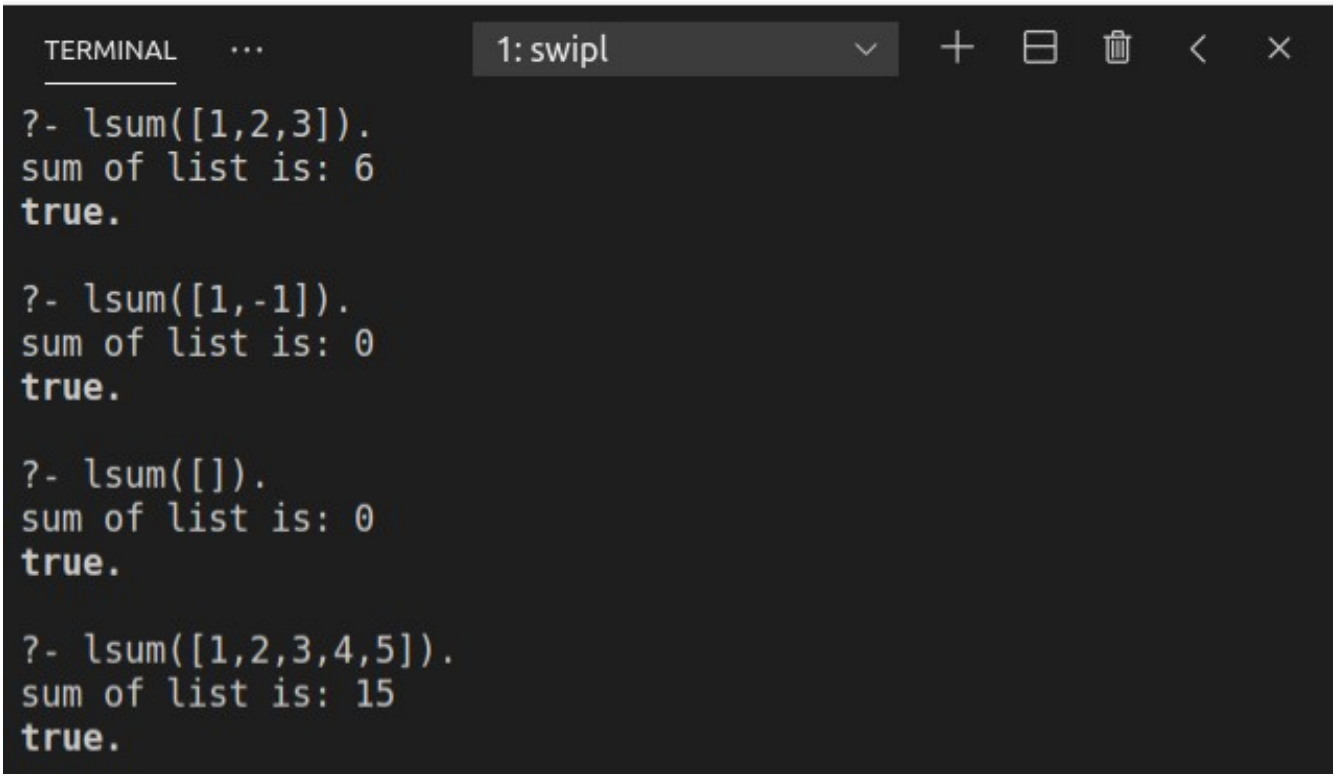

Figure 14: Solution 14 Image

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)
).
```

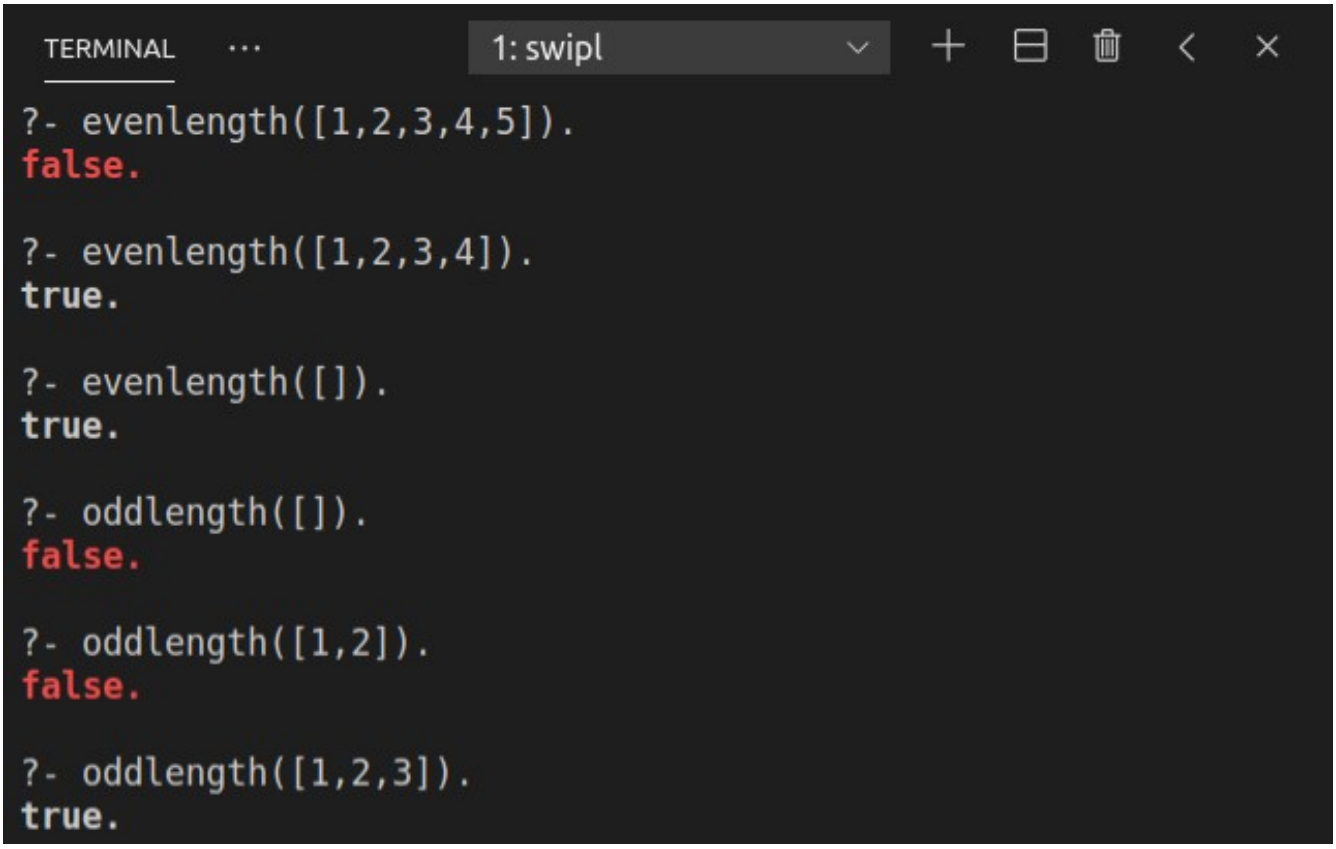

Figure 15: Solution 15 Image

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")
).
```

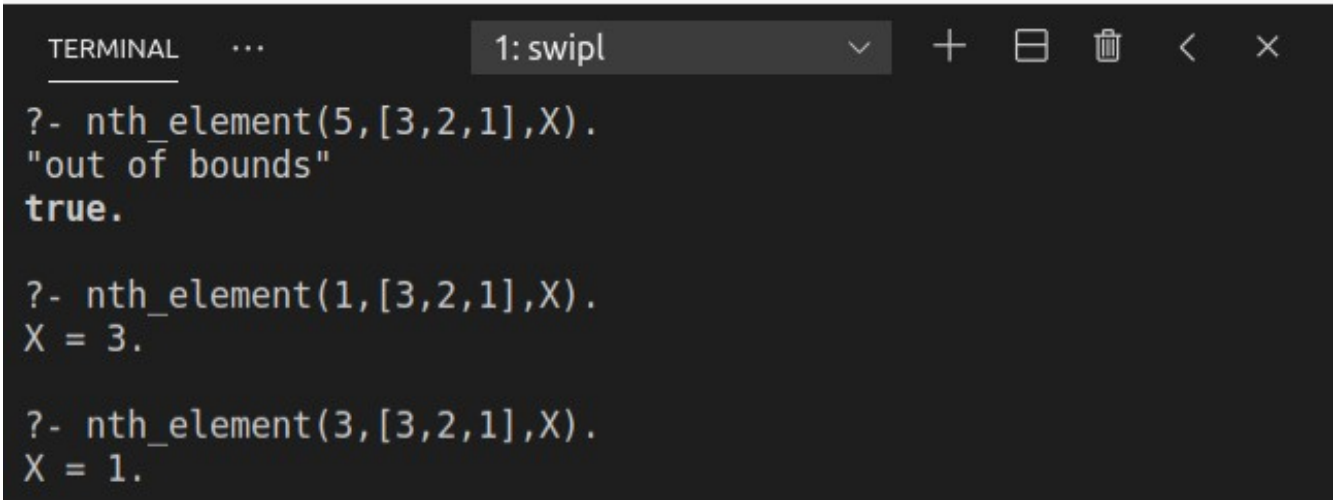

Figure 16: Solution 16 Image

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)
).
```

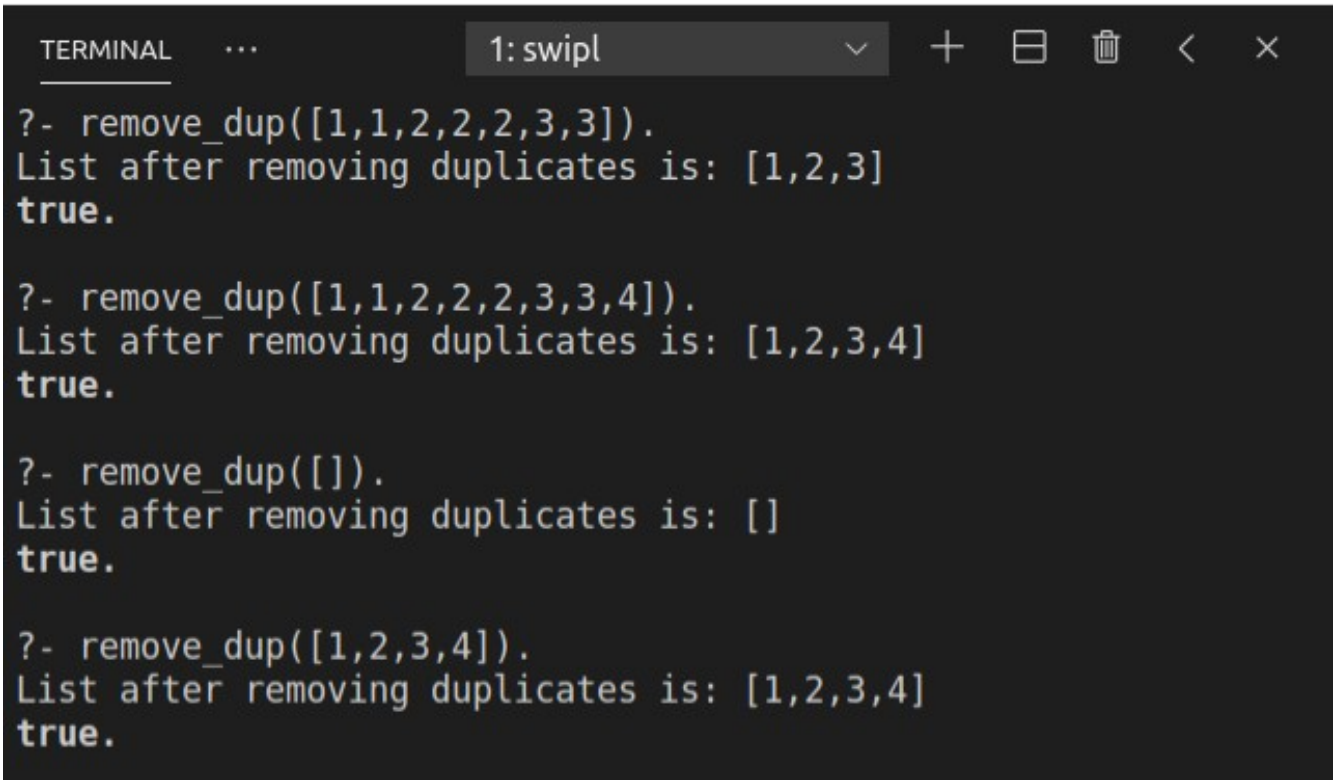

Figure 17: Solution 17 Image

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)
).
```

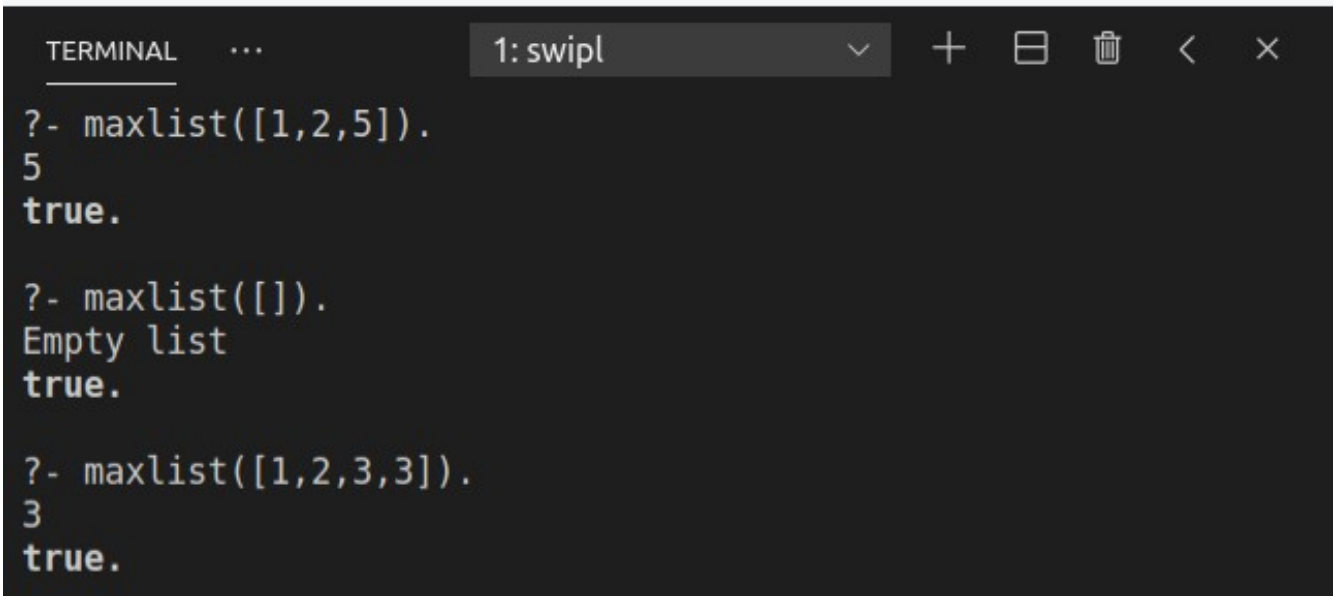

Figure 18: Solution 18 Image

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)
).
```

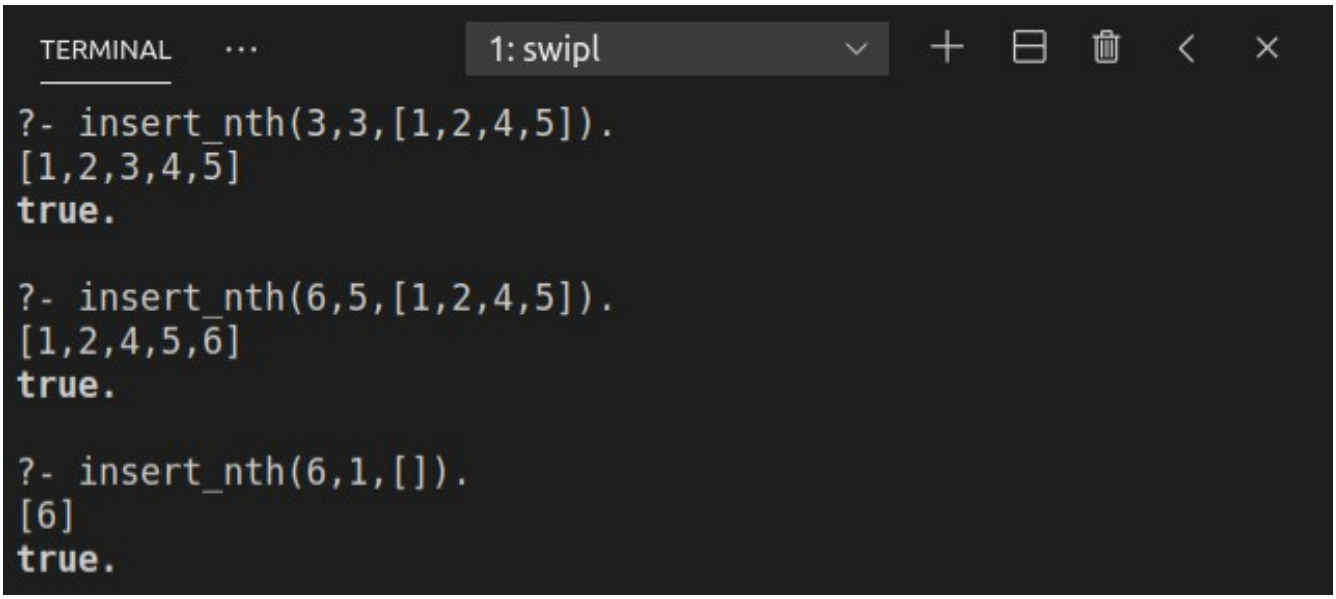

Figure 19: Solution 19 Image

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)
).
```

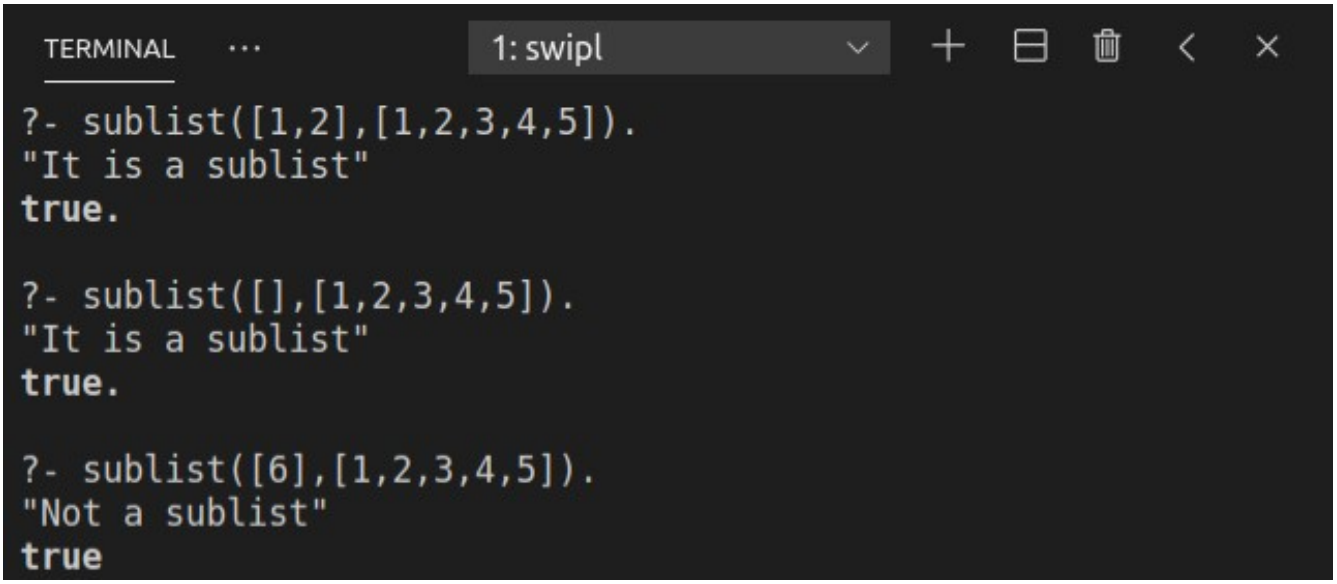

Figure 20: Solution 20 Image

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)
).
```

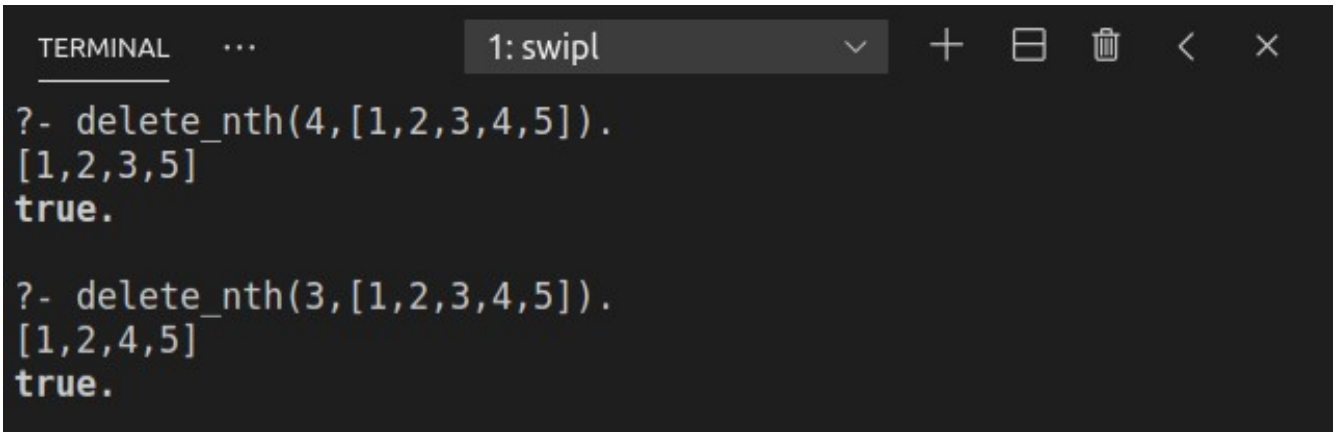

Figure 21: Solution 21 Image

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).
```

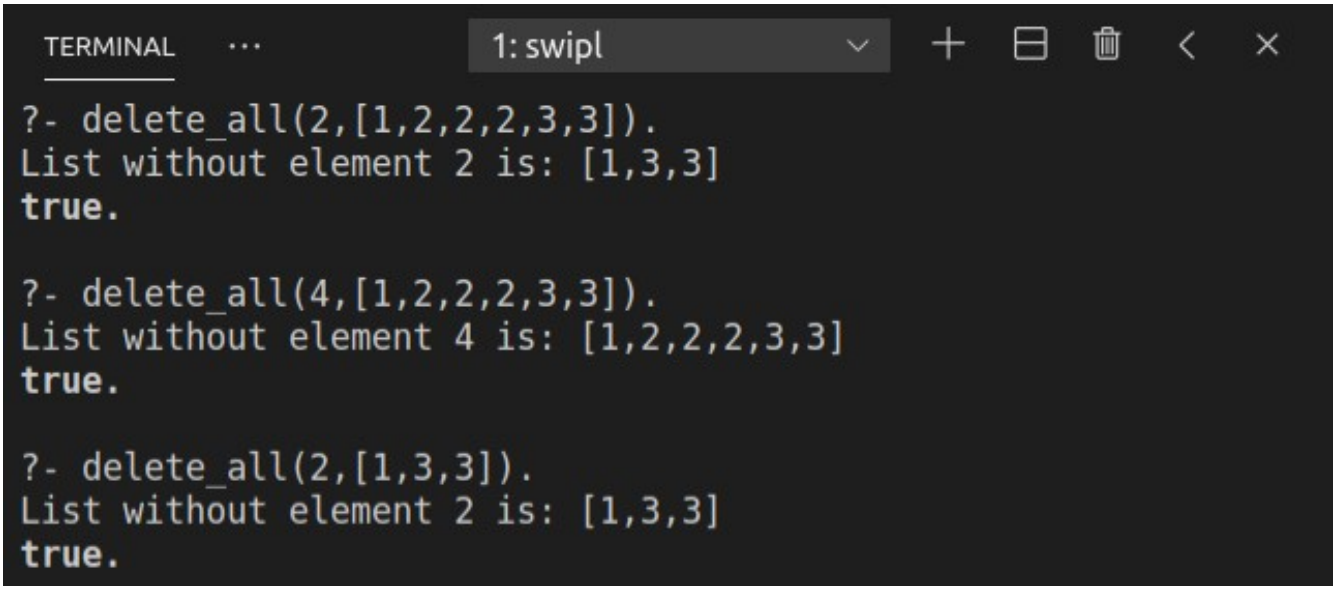

Figure 22: Solution 22 Image

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)
).
```

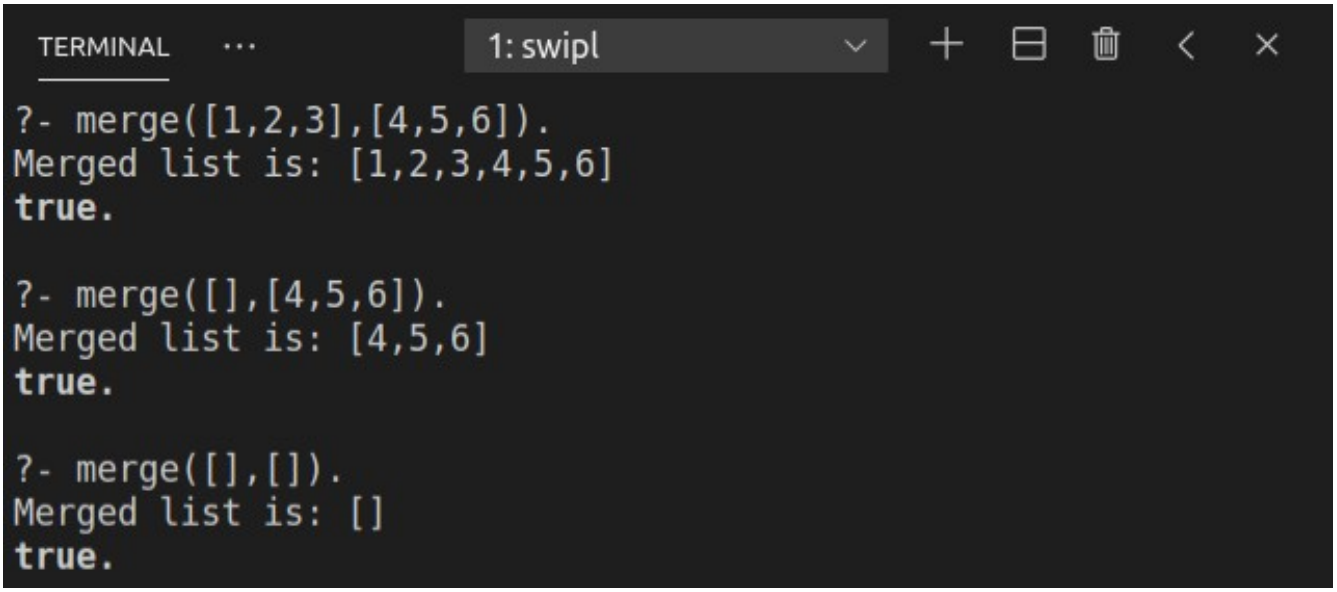

Figure 23: Solution 23 Image