Practical File

Subject: Artificial Intelligence

Submitted to:
Dr. Ankit Rajpal
Assistant Professor(University Of Delhi)

Submitted by:
Himanshu Madan 17HCS4116

List Of Practicals

S. NO	Particulars	Date	Teacher's Signature/ Remarks
1.	Write a prolog program to calculate the sum of two numbers.		
2.	Write a Prolog program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y.		
3.	Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N.		
4.	Write a program in PROLOG to implement generate_fib(N,T) where T represents the Nth term of the fibonacci series.		
5.	Write a Prolog program to implement GCD of two numbers.		
6.	Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.		
7.	Prolog program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.		
8.	Write a program in PROLOG to implement towerofhanoi (N) where N represents the number of discs		
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.		
10	Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not.		
11.	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.		
12.	Write a Prolog program to implement reverse (L, R) where List L is original and List R is reversed list.		
13.	Write a program in PROLOG to implement palindrome (L) which checks whether a list L is a palindrome or not.		
14.	Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.		
15.	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		
16.	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.		
17.	Write a program in PROLOG to implement remove_dup (L, R) where L denotes the list with some duplicates and the list R denotes the list with duplicates removed.		
18.	Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list		

19.	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.		
20.	Write a Program in PROLOG to implement sub-list(S, L) that checks whether the list S is the sublist of list L or not. (Check for sequence or the part in the same order).		
21.	Write a Prolog program to implement delete_nth (N, L, R) that removes the element on Nth posi-tion from a list L to generate a list R.		
22.	Write a program in PROLOG to implement delete_all (X, L, R) where X denotes the element whose all occurrences has to be deleted from list L to obtain list R.		
23.	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.		
24.	Write a PROLOG program that will take grammar rules in the following format: NT → (NT | T)* Where NT is any nonterminal, T is any terminal and Kleene star (*) signifies any number of repetitions, and generate the corresponding top-down parser, that is: sentence -> noun-phrase, verb-phrase determiner -> [the] will generate the following: sentence (I, O) :- noun-phrase(I,R), verb-phrase (R,O). determiner ([the|X], X) :- !.		

25.	Write a prolog program that implements Semantic Networks (ATN/RTN).		
	Extended List of Practicals:-		
26.	Write a prolog program to find whether a given list is a subsequence of another list or not.		
27.	Write a prolog program to find the sum of the digits of a number.		
28.	Write a prolog program to find the last element of the list.		
29.	Write a prolog program to find length of the list using tail recursion.		
30.	Write a prolog program to print the INORDER, POSTORDER and PREORDER traversal of a tree.		
31.	Write a prolog program to reverse a list using tail recursion.		
32.	Write a prolog program to check whether a list is a permutation of another list or not.		

Practical 1

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

```
go :-   write('Enter the first number:'),

   read(X1),
   write('Enter the second number:'),
   read(X2),
   sum(X1,X2,R),nl,
   write('The sum is: '),write(R).

   sum(A,B,R):-R is A + B.

:-initialization(go).
```

Output :-

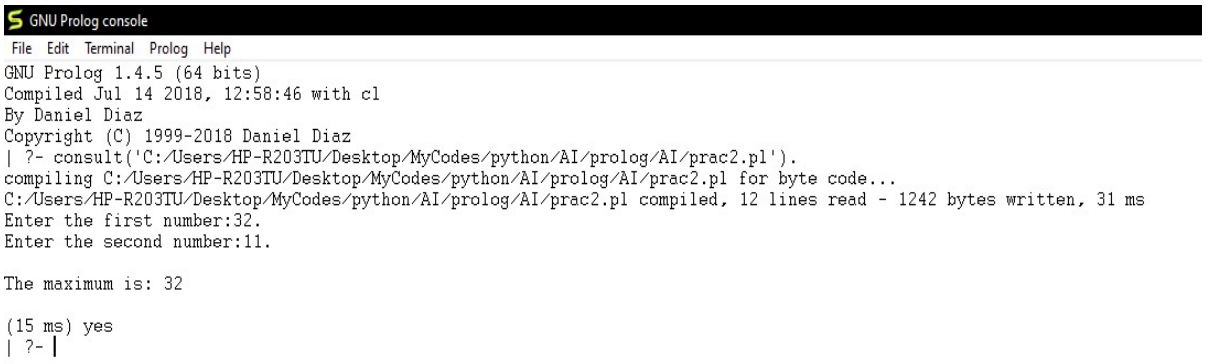
```
GNU Prolog console
File  Edit  Terminal  Prolog  Help
GNU Prolog 1.4.5 (64 bits)
Compiled Jul 14 2018, 12:58:46 with cl
By Daniel Diaz
Copyright (C) 1999-2018 Daniel Diaz
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac1.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac1.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac1.pl compiled, 11 lines read - 1169 bytes written, 15 ms
Enter the first number:21.
Enter the second number:14.

The sum is: 35

yes
| ?-
```

Practical 2

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

```
go :-   write('Enter the first number:'),

   read(X1),
   write('Enter the second number:'),
   read(X2),
   mymax(X1,X2,R),nl,
   write('The maximum is: '),write(R).

 mymax(X,Y,X):- X > Y,!.

 mymax(_,Y,Y).

:-initialization(go).
```

Output:-



Practical 3

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

```
go :-   write('Enter the number:'),

   read(X1),
   factorial(X1,R),nl,
   write('The factorial is: '),write(R).

   factorial(0,1).

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

:-initialization(go).
```

Output:-

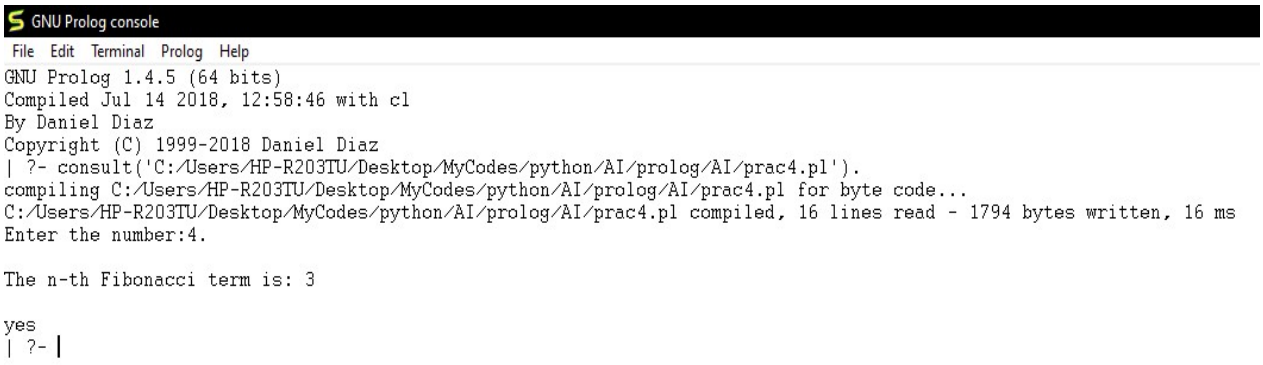
```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac3.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac3.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac3.pl compiled, 13 lines read - 1382 bytes written, 46 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac3.pl:1: redefining procedure go/0
        C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac2.pl:1: previous definition
Enter the number:14.

The factorial is: 87178291200

yes
| ?-
```

Practical 4

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

```
go :-   write('Enter the number:'),

   read(X1),
   generate_fib(X1,R),nl,
   write('The n-th Fibonacci term is: '),write(R).

   generate_fib(0,0).
   generate_fib(0,1).

   generate_fib(N,X):- N1 is N - 1,
         generate_fib(N1,X1),
         N2 is N - 2,
         generate_fib(N2,X2),
         X is X1 + X2.

:-initialization(go).
```

Output:-



Practical 5

Ques: Write a Prolog program to implement GCD of two numbers.

```
go :-   write('Enter the first number:'),

   read(X1),
   write('Enter the second number:'),
   read(X2),
   gcd(X1,X2,R),nl,
   write('The GCD is: '),write(R).


 gcd(0,B,B).
 gcd(A,0,A).
 gcd(A,A,A).

 gcd(A,B,R):-  A > B,
        N1 is A - B,
        gcd(N1,B,R);
        N1 is B - A,
        gcd(A,N1,R).


:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac5.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac5.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac5.pl compiled, 20 lines read - 2308 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac5.pl:1: redefining procedure go/0
         C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac4.pl:1: previous definition
Enter the first number:11.
Enter the second number:5.

The GCD is: 1

(32 ms) yes
| ?-
```

Practical 6

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

```
go :-   write('Enter the number:'),

   read(X1),
   write('Enter the exponent:'),
   read(X2),
   exp(X1,X2,R),nl,
   write('The answer is: '),write(R).


  exp(Num,1,Num).

  exp(Num,Pow,Ans):- Pow1 is Pow - 1,
          exp(Num,Pow1,Ans1),
          Ans is Ans1 * Num.


:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac6.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac6.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac6.pl compiled, 16 lines read - 1568 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac6.pl:1: redefining procedure go/0
         C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac5.pl:1: previous definition
Enter the number:8.
Enter the exponent:3.

The answer is: 512

(16 ms) yes
| ?-
```

Practical 7

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

```
go :-   write('Enter the first number:'),
   read(X1),
   write('Enter the second number:'),
   read(X2),
   prod(X1,X2,R),nl,
   write('The product is: '),write(R).


 mult(X, 0, 0).
 mult(X, Y, R):- Y1 is Y - 1,
        mult(X, Y1, R1),
        R is R1 + X.
 prod(X, Y, R):- Y < 0,
        Y1 is -1 * Y,
        mult(X, Y1, R1),
        R is -1 * R1;
        mult(X, Y, R).

:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl:9: warning: singleton variables [X] for mult/3
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl compiled, 19 lines read - 2702 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl:1: redefining procedure go/0
        C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac6.pl:1: previous definition
Enter the first number:10.
Enter the second number:4.

The product is: 40

(47 ms) yes
| ?-
```

Practical 8

Ques: Write a program in PROLOG to implement towerofhanoi (N) where N represents the number of discs.

```
go:- write('Enter number of discs: '),

  read(N),
  towerOfHanoi(N, a, c, b).

towerOfHanoi(1, R1, R2, R3):- write('Move disk 1 from rod '),
          write(R1),
          write(' to rod '),
          write(R2), nl.
towerOfHanoi(N, R1, R2, R3):-
        K is N - 1,
        towerOfHanoi(K, R1, R3, R2),
        write('Move disk '),
        write(N),
        write(' from rod '),
        write(R1),
        write(' to rod '),
        write(R2), nl,
        towerOfHanoi(K, R3, R2, R1).
:- initialization(go).
```

Output:-

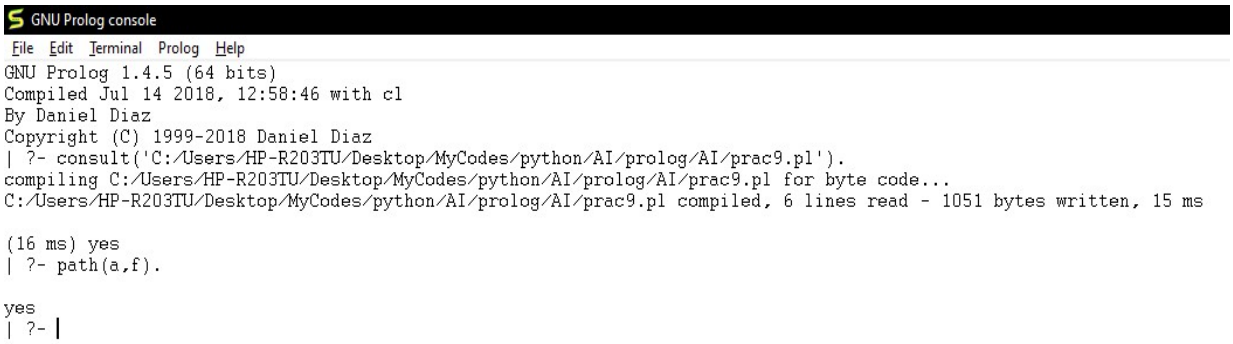
```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl:5-8: warning: singleton variables [R3] for towerOfHanoi/4
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl compiled, 19 lines read - 2178 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl:1: redefining procedure go/0
         C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac7.pl:1: previous definition
Enter number of discs: 4.
Move disk 1 from rod a to rod b
Move disk 2 from rod a to rod c
Move disk 1 from rod b to rod c
Move disk 3 from rod a to rod b
Move disk 1 from rod c to rod a
Move disk 2 from rod c to rod b
Move disk 1 from rod a to rod b
Move disk 4 from rod a to rod c
Move disk 1 from rod b to rod c
Move disk 2 from rod b to rod a
Move disk 1 from rod c to rod a
Move disk 3 from rod b to rod c
Move disk 1 from rod a to rod b
Move disk 2 from rod a to rod c
Move disk 1 from rod b to rod c

yes
| ?-
```

Practical 9

Ques: 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.

```
edge(a,b).

edge(b,c).
edge(c,d).
edge(d,e).
edge(e,f).
path(X,Y):- edge(X,Y),!.
path(X,Y):- edge(X,Z),path(Z,Y).
```

Output:-



Practical 10

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

```
go :-   write('Enter the List(-1 to end)'),nl,


   createList(L),
   write('List: '),
   printList(L),nl,
   write('Enter the element to be searched for:'),
   read(X),
   is_member(X,L),write('Yes,it is a member');
   write('Not a member').

   enterElement(X):- write('Enter element: '),
            read(X).

   createList(L):- enterElement(X),
           createListHelper(X, L).
   createListHelper(-1, []):- !.
   createListHelper(X, [X|Y]):- enterElement(X1),
               createListHelper(X1, Y).

   printList([]):- !.
   printList([X|Y]):-  write(X),
            write(' '),
            printList(Y).

   is_member(H,[H|_]):- !
   is_member(H,[_|T]):- is_member(H,T).

:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac10.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac10.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac10.pl compiled, 26 lines read - 3465 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac10.pl:1: redefining procedure go/0
         C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac8.pl:1: previous definition
Enter the List(-1 to end)
Enter element: 2.
Enter element: 1.
Enter element: 4.
Enter element: 2.
Enter element: -1.
List: 2 1 4 2
Enter the element to be searched for:4.
Yes,it is a member

(47 ms) yes
| ?-
```

Practical 11

Ques: 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.

```
go :-   write('Enter the first List(-1 to end)'),nl,


   createList(X1),
   write('List one: '),
   printList(X1),nl,
   write('Enter the second List(-1 to end)'),nl,
   createList(X2),
   write('List two: '),
   printList(X2),nl,
   conc(X1,X2,X3),nl,
   write('The new list is: '),
   printList(X3).

   enterElement(X):- write('Enter element: '),
            read(X).

   createList(L):- enterElement(X),
          createListHelper(X, L).
   createListHelper(-1, []):- !.
   createListHelper(X, [X|Y]):- enterElement(X1),
               createListHelper(X1, Y).

   printList([]):- !.
   printList([X|Y]):-  write(X),
            write(' '),
            printList(Y).

   conc([],L2,L2).
   conc([H|T1],L2,[H|T2]):- conc(T1,L2,T2).

:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac11.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac11.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac11.pl compiled, 29 lines read - 3316 bytes written, 46 ms
Enter the first List(-1 to end)
Enter element: 7.
Enter element: 8.
Enter element: -1.
List one: 7 8
Enter the second List(-1 to end)
Enter element: 5.
Enter element: 8.
Enter element: 2.
Enter element: -1.
List two: 5 8 2

The new list is: 7 8 5 8 2

(187 ms) yes
| ?-
```

Practical 12

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

```
go:- write('Enter List(-1 to end)'),nl,

  createList(L),
  write('List: '),
  printList(L),nl,
  maxOfList(L, M),
  write('Maximum element of the list is: '),
  write(M), nl.

enterElement(X):- write('Enter the new element: '),
        read(X).

createList(L):- enterElement(X),
       createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
          createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

my_max(X, Y, M):- X > Y, M is X;
        M is Y.
maxOfListHelper([], Current, Current).
maxOfListHelper([H|T], Previous, M):- my_max(H, Previous, Cur-
rent), maxOfListHelper(T, Current, M).
maxOfList([H|T], M):- maxOfListHelper(T, H, M).

:- initialization(go).
```

Output:-

```
Enter the List(-1 to end)
Enter element: 3.
Enter element: 1.
Enter element: 5.
Enter element: -1.
List: 3 1 5
The reversed list is: 5 1 3

(63 ms) yes
| ?-
```

Practical 13

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

```
go :-   write('Enter the List(-1 to end)'),nl,

   createList(L),
   write('List: '),
   printList(L),nl,
   my_reverse(L,L),
   write('Yes,it is a palindrome.'),nl;
   write('Not a palindrome.'),nl.

   enterElement(X):- write('Enter element: '),
        read(X).

   createList(L):- enterElement(X),
          createListHelper(X, L).
   createListHelper(-1, []):- !.
   createListHelper(X, [X|Y]):- enterElement(X1),
              createListHelper(X1, Y).

   printList([]):- !.
   printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

   reverseHelper([], A, A):- !.
   reverseHelper([X|Y], R, A):- reverseHelper(Y, R, [X|A]).
   my_reverse(L, R):- reverseHelper(L, R, []).

:- initialization(go).
```

Output:-

```
Enter the List(-1 to end)
Enter element: 5.
Enter element: 8.
Enter element: 3.
Enter element: -1.
List: 5 8 3
Not a palindrome.

(140 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac13.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac13.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac13.pl compiled, 27 lines read - 3613 bytes written, 16 ms
Enter the List(-1 to end)
Enter element: 1.
Enter element: 2.
Enter element: 1.
Enter element: -1.
List: 1 2 1
Yes,it is a palindrome.

(16 ms) yes
| ?-
```

Practical 14

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

```
go :-   write('Enter the List(-1 to end)'),nl,

   createList(L),
   write('List you entered is: '),
   printList(L),nl,
   write('Sum of the list is : '),
   list_sum(L,R),
   write(R).


   enterElement(X):- write('Enter element: '),
            read(X).

   createList(L):- enterElement(X),
          createListHelper(X, L).
   createListHelper(-1, []):- !.
   createListHelper(X, [X|Y]):- enterElement(X1),
               createListHelper(X1, Y).

   printList([]):- !.
   printList([X|Y]):-  write(X),
            write(' '),
            printList(Y).

   list_sum([],0):- !.
   list_sum([H|T],R):- list_sum(T,X),
            R is H + X.

:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac14.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac14.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac14.pl compiled, 27 lines read - 3083 bytes written, 31 ms
Enter the List(-1 to end)
Enter element: 3.
Enter element: 4.
Enter element: 2.
Enter element: -1.
List you entered is: 3 4 2
Sum of the list is : 9

(47 ms) yes
| ?-
```

Practical 15

Ques: 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.

```
go:- write('Enter List(-1 to specify end)'),


  nl,
  createList(L),
  write('List: '),
  printList(L),
  nl,
  oddlength(L),
  write('The list is of odd length');
  write('The list is of even length').

enterElement(X):- write('Enter element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]):- !.
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).
evenlength([]).
evenlength([_|Y]):- oddlength(Y).
oddlength([X|[]]).
oddlength([_|Y]):- evenlength(Y).
:- initialization(go).
```

Output:-

```
(47 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl:26: warning: singleton variables [X] for oddlength/1
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl compiled, 27 lines read - 3482 bytes written, 45 ms
Enter List(-1 to specify end)
Enter element: 3.
Enter element: 2.
Enter element: 1.
Enter element: -1.
List: 3 2 1
The list is of odd length

(47 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl:26: warning: singleton variables [X] for oddlength/1
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac15.pl compiled, 27 lines read - 3482 bytes written, 15 ms
Enter List(-1 to specify end)
Enter element: 4.
Enter element: 2.
Enter element: 8.
Enter element: 9.
Enter element: -1.
List: 4 2 8 9
The list is of even length

(16 ms) yes
| ?-
```

Practical 16

Ques: 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.

```
go:- write('Enter List(-1 to specify end)'),


  nl,
  createList(L),
  write('List: '),
  printList(L),
  nl,
  write('Enter N:'),nl,
  read(N),
  write('The element is:'),
  nth_element(L,N,R),nl,
  write(R).

  enterElement(X):- write('Enter element: '),
        read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
         createListHelper(X1, Y).

printList([]):- !.
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

   nth_element([H|T],1,H):- !.

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

Output:-

```
Enter List(-1 to specify end)
Enter element: 3.
Enter element: 2.
Enter element: 6.
Enter element: 1.
Enter element: -1
.
List: 3 2 6 1
Enter N:
3.
The element is:
6

(46 ms) yes
| ?-
```

Practical 17

Ques: Write a program in PROLOG to implement remove_dup (L, R) where L denotes the list with some duplicates and the list R denotes the list with duplicates removed.

```
go:- write('Enter List(-1 to end)'), nl,

 createList(L),
 write('List: '),
 printList(L),
 nl,
 remove_dup(L, R),
 write('List after removing duplicates: '),
 printList(R), nl.

enterElement(X):- write('Enter element: '),
      read(X).

createList(L):- enterElement(X),
     createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

is_member(X, [X|_]).
is_member(X, [_|Y]):- is_member(X, Y).

remove_dup([], []).
remove_dup([X|Y], R):- is_member(X, Y),
        remove_dup(Y, R).
remove_dup([X|Y], [X|R]):- \+is_member(X, Y),
          remove_dup(Y, R).
:- initialization(go).
```

Output:-

```
(31 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac17.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac17.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac17.pl compiled, 33 lines read - 3759 bytes written, 46 ms
Enter List(-1 to end)
Enter element: 4.
Enter element: 3.
Enter element: 2.
Enter element: 3.
Enter element: -1.
List: 4 3 2 3
List after removing duplicates: 4 2 3

yes
| ?-
```

Practical 18

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

```
go:- write('Enter List(-1 to end)'),nl,

 createList(L),
 write('List: '),
 printList(L),nl,
 maxOfList(L, M),
 write('Maximum element of the list is: '),
 write(M), nl.

enterElement(X):- write('Enter the new element: '),
      read(X).

createList(L):- enterElement(X),
     createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
       createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
      write(' '),
      printList(Y).

my_max(X, Y, M):- X > Y, M is X;
      M is Y.
maxOfListHelper([], Current, Current).
maxOfListHelper([H|T], Previous, M):- my_max(H, Previous, Cur-
rent), maxOfListHelper(T, Current, M).
maxOfList([H|T], M):- maxOfListHelper(T, H, M).

:- initialization(go).
```

Output:-

```
        C:/Users/HP-RZU3IU/Desktop/MyCodes/python/AI/prolog/AI/prac17.pl:20: previous definition
Enter List(-1 to end)
Enter the new element: 3.
Enter the new element: 1.
Enter the new element: -1.
List: 3 1
Maximum element of the list is: 3

(78 ms) yes
| ?-
```

Practical 19

Ques: 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.

```
go:- write('Enter List(-1 to end)'),nl,


  createList(L),
  write('List: '),
  printList(L),nl,
  print("Enter the value of n:"),nl,
  read(N),nl,
  print("Enter the element to be inserted:"),nl,
  read(I),nl,
  insert_nth(I,N,L,R),
  write('The new list is: '),
  printList(R), nl.

enterElement(X):- write('Enter the new element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

conc([], L2, L2).
conc([X|T2], L2, [X|T1]):- conc(T1, L2, T2).

insert_nth(I, 1, L, R):- conc([I], L, R), !.
insert_nth(I, N, [X|T1], [X|T2]):- N1 is N - 1,
            insert_nth(I, N1, T1, T2).
:- initialization(go).
```

Output:-

```
(62 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac19.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac19.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac19.pl compiled, 33 lines read - 4293 bytes written, 46 ms
Enter List(-1 to end)
Enter the new element: 3.
Enter the new element: 2.
Enter the new element: 6.
Enter the new element: 9.
Enter the new element: -1.
List: 3 2 6 9
Enter the value of n:
2.

Enter the element to be inserted:
11.
The new list is: 3 11 2 6 9

(31 ms) yes
| ?-
```

Practical 20

Ques: Write a Program in PROLOG to implement sublist(S, L) that checks whether the list S is the sublist of list L or not. (Check for sequence or the part in the same order).

```
go:- write('Enter List(-1 to end)'),nl,


  createList(L),
  write('List: '),
  printList(L),nl,
  write('Enter the list to be checked'),
  createList(S),
  subList(S,L),
  write('Yes'),nl;
  write('No').

enterElement(X):- write('Enter the new element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

conc([], L2, L2).
conc([X|T1], L2, [X|T2]):- conc(T1, L2, T2).

subList(S, L):- conc(L1, L2, L),
      conc(S, L3, L2).

:- initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac20.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac20.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac20.pl:28-29: warning: singleton variables [L1,L3] for subList/2
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac20.pl compiled, 31 lines read - 3757 bytes written, 31 ms
Enter List(-1 to end)
Enter the new element: 3.
Enter the new element: 2.
Enter the new element: 5.
Enter the new element: -1.
List: 3 2 5
Enter the list to be checkedEnter the new element: 2.
Enter the new element: 5.
Enter the new element: -1.
Yes

(78 ms) yes
| ?-
```

Practical 21

Ques: 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.

```
go:- write('Enter List(-1 to end)'),nl,


 createList(L),
 write('List: '),
 printList(L),nl,
 print("Enter the value of n:"),nl,
 read(N),nl,
 delete_nth(N,L,R),
 write('The new list is: '),
 printList(R), nl.

enterElement(X):- write('Enter the new element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

delete_nth(1,[H|T],T):- !.
delete_nth(N,[X|Y],[X|Z]):- N1 is N - 1,
          delete_nth(N1,Y,Z).

:- initialization(go).
```

Output:-

```
Enter List(-1 to end)
Enter the new element: 3.
Enter the new element: 5
.
Enter the new element: 4.
Enter the new element: 2.
Enter the new element: -1.
List: 3 5 4 2
Enter the value of n:
3.

The new list is: 3 5 2

(47 ms) yes
| ?-
```

Practical 22

Ques: Write a program in PROLOG to implement delete_all (X, L, R) where X denotes the element whose all occurrences has to be deleted from list L to obtain list R.

```
go:- write('Enter List(-1 to end)'),nl,

 createList(L),
 write('List: '),
 printList(L),nl,
 print("Enter the element to be deleted:"),nl,
 read(N),nl,
 delete_all(N,L,R),
 write('The new list is: '),
 printList(R), nl.

enterElement(X):- write('Enter the new element: '),
      read(X).

createList(L):- enterElement(X),
     createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
       createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

delete_all(X, [], []).
delete_all(X, [H|T], Z):-  X = H, delete_all(X, T, Z), !.
delete_all(X, [H|T], [H|Z]):- X \= H, delete_all(X, T, Z).

:- initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac22.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac22.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac22.pl:25: warning: singleton variables [X] for delete_all/3
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac22.pl compiled, 29 lines read - 3624 bytes written, 15 ms
Enter List(-1 to end)
Enter the new element: 3.
Enter the new element: 6.
Enter the new element: 4.
Enter the new element: 3.
Enter the new element: -1.
List: 3 6 4 3
Enter the element to be deleted:
3.

The new list is: 6 4

(15 ms) yes
| ?-
```

Practical 23

Ques: 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.

```
go:- write('Enter first sorted list(-1 to end)'),nl,


  createList(L1),
  write('List: '),
  printList(L1),nl,
  write('Enter the second sorted list'),nl,
  createList(L2),nl,
  printList(L2),nl,
  write('The new merged list is: '),
  merge(L1,L2,R),nl,
  printList(R), nl.

enterElement(X):- write('Enter the new element: '),
        read(X).

createList(L):- enterElement(X),
       createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
         createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

merge([],L2,L2).
merge(L1,[],L1).
merge([H1|T1],[H2|T2],[H1|Z]):- H1 =< H2,merge(T1,[H2|T2],Z),!.
merge([H1|T1],[H2|T2],[H2|Z]):- H1 > H2,merge([H1|T1],T2,Z).

:- initialization(go).
```

Output:-

```
(47 ms) yes
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac23.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac23.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac23.pl compiled, 32 lines read - 4200 bytes written, 31 ms
Enter first sorted list(-1 to end)
Enter the new element: 3.
Enter the new element: 5.
Enter the new element: 7.
Enter the new element: -1.
List: 3 5 7
Enter the second sorted list
Enter the new element: 1.
Enter the new element: 2.
Enter the new element: 4.
Enter the new element: -1.

1 2 4
The new merged list is:
1 2 3 4 5 7

(47 ms) yes
| ?-
```

Practical 26

Ques: Write a prolog program to find whether a given list is a subsequence of another list or not.

```
go:- write('Enter List(-1 to end)'),nl,

  createList(L),
  write('List: '),
  printList(L),nl,
  write('Enter the list to be checked: '),
  createList(S),
  is_subsequence(S,L),
  write('Yes, it is a subsequence'),nl;
  write('Not a subsequence').

is_subsequence([],[_|_]).
is_subsequence([H|T],[H|T1]):- is_subsequence(T,T1),!.
is_subsequence([H|T],[H1|T1]):- is_subsequence([H|T],T1),!.

:- initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl:13: warning: singleton variables [H1] for is_subsequence/2
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl compiled, 14 lines read - 2506 bytes written, 31 ms
Enter List(-1 to end)
Enter the new element: 2.
Enter the new element: 3.
Enter the new element: 4.
Enter the new element: -1.
List: 2 3 4
Enter the list to be checked: Enter the new element: 2.
Enter the new element: 3.
Enter the new element: -1.
Yes, it is a subsequence

(63 ms) yes
| ?-
```

Practical 27

Ques: Write a prolog program to find the sum of the digits of a number.

```
go :-   write('Enter the number:'),


   read(X1),
   sum_of_digits(X1,R),nl,
   write('The sum of digits of the number is: '),write(R).

   sum_of_digits(0,0).
   sum_of_digits(A,R):- X is A mod 10,
         A1 is A // 10,
         sum_of_digits(A1,R2),
         R is R2 + X.

:-initialization(go).
```

Output:-

```
| ?- consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac27.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac27.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac27.pl compiled, 12 lines read - 1599 bytes written, 31 ms
warning: C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac27.pl:1: redefining procedure go/0
         C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl:1: previous definition
Enter the number:34.

The sum of digits of the number is: 7

(78 ms) yes
| ?-
```

Practical 28

Ques: Write a prolog program to find the last element of the list.

```
go:- write('Enter List(-1 to end)'),

 nl,
 createList(L),
 write('List: '),
 printList(L),nl,
 last_ele(L, R),
 write('The last element of the list is: '),
 write(R), nl.

enterElement(X):- write('Enter element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

last_ele([H],H).
last_ele([_|T],R):- last_ele(T,R).

:- initialization(go).
```

Output:-

```
C:/Users/HP R2510/Desktop/MyCodes/Python/AI/Prolog/AI/prac23.pl:21: previous definition
Enter List(-1 to end)
Enter element: 3
.
Enter element: 2.
Enter element: 5.
Enter element: 1.
Enter element: -1.
List: 3 2 5 1
The last element of the list is: 1

(31 ms) yes
| ?-
```

Practical 29

Ques: Write a prolog program to find length of the list using tail recursion.

```
go:- write('Enter List(-1 to end)'),

  nl,
  createList(L),
  write('List: '),
  printList(L),nl,
  list_length(L, R),
  write('The length of the list is: '),
  write(R), nl.

enterElement(X):- write('Enter element: '),
       read(X).

createList(L):- enterElement(X),
      createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
        createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
       write(' '),
       printList(Y).

list_length(X,L) :- list_length_helper(X,0,L) .

list_length_helper([],L,L) .
list_length_helper([_|X],T,L) :- T1 is T + 1,
           list_length_helper(X,T1,L).

:- initialization(go).
```

Output:-

```
        C:/Users/MF-K20310/Desktop/MyCodes/python/AI/prolog/AI/prac26.pl:19: previous definition
Enter List(-1 to end)
Enter element: 3.
Enter element: 4.
Enter element: 2.
Enter element: 5.
Enter element: -1.
List: 3 4 2 5
The length of the list is: 4

(62 ms) yes
| ?-
```

Practical 30

Ques: Write a prolog program to print the INORDER, POSTORDER and PREORDER traversal of a tree.

```
go:- write('Enter the Tree : '),

 read(T), nl,

 write('The Pre-order traversal is  : '),

 preorder(T,R1), write(R1),nl.
 write('The In-order traversal is  : '),
 inorder(T,R2), write(R2),nl.
 write('The Post-order traversal is  : '),
 postorder(T,R3), write(R3),nl.

preorder(nil,[]):-!.
preorder(tr(nil,X,nil),[X]):-!.
preorder(tr(LEFT,R,RIGHT),L):-
      preorder(LEFT,LT), preorder(RIGHT,RT),
      append([R],LT,Temp),
      append(Temp,RT,L),!.

inorder(nil,[]):-!.
inorder(tr(nil,X,nil),[X]):-!.
inorder(tr(LEFT,R,RIGHT),L):-
     inorder(LEFT,LT), inorder(RIGHT,RT),
     append(LT,[R],Temp),
     append(Temp,RT,L),!.

postorder(nil,[]):-!.
postorder(tr(nil,X,nil),[X]):-!.
postorder(tr(LEFT,R,RIGHT),L):-
      postorder(LEFT,LT), postorder(RIGHT,RT),
      append(LT,RT,Temp),
      append(Temp,[R],L),!.
:- initialization(go).
```

Output:-

```
consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac30.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac30.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac30.pl compiled, 31 lines read - 4608 bytes written, 46 ms
Enter the Tree : tr(tr(tr(nil,48,tr(nil,49,nil)),50,nil),68,tr(tr(nil,75,nil),77,tr(nil,79,nil)))
.

The Pre-order traversal is  : [68,50,48,49,77,75,79]
The In-order traversal is  : [48,49,50,68,75,77,79]
The Post-order traversal is  : [49,48,50,75,79,77,68]

(141 ms) yes
| ?-
```

Practical 31

Ques: Write a prolog program to reverse a list using tail recursion.

```
go:- write('Enter List(-1 to end)'),

  nl,
  createList(L),
  write('List: '),
  printList(L),nl,
  reverse_list(L, R),
  write('The reverse of the list is: '),
  printList(R), nl.

enterElement(X):- write('Enter element: '),
        read(X).

createList(L):- enterElement(X),
       createListHelper(X, L).
createListHelper(-1, []):- !.
createListHelper(X, [X|Y]):- enterElement(X1),
         createListHelper(X1, Y).

printList([]).
printList([X|Y]):-  write(X),
        write(' '),
        printList(Y).

reverse_list(L,R):- reverse_list_helper(L,[],R).
reverse_list_helper([],A,A):- !.
reverse_list_helper([H|T],A1,A2):- reverse_list_helper(T, [H|A1],
A2).

:- initialization(go).
```

Output:-

```
        C:/Users/hp-k20310/Desktop/MyCodes/python/AI/prolog/AI/prac29.pl:19: previous definition
Enter List(-1 to end)
Enter element: 3.
Enter element: 4.
Enter element: 2.
Enter element: 6.
Enter element: -1.
List: 3 4 2 6
The reverse of the list is: 6 2 4 3

(78 ms) yes
| ?-
```

Practical 32

Ques: Write a prolog program to check whether a list is a permutation of another list or not.

```
conc([], L2, L2).


conc([X|Y], L2, [X|Z]):- conc(Y, L2, Z).

my_permutation([], []).
my_permutation(L, [H|T]):- conc(V, [H|U], L),
          conc(V, U, W),
          my_permutation(W, T).
```

Output:-

```
consult('C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac32.pl').
compiling C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac32.pl for byte code...
C:/Users/HP-R203TU/Desktop/MyCodes/python/AI/prolog/AI/prac32.pl compiled, 6 lines read - 1200 bytes written, 31 ms

(15 ms) yes
| ?- my_permutation([1,3,2],[3,1,2]).

true ?

yes
| ?-
```

****