Roll No :	**1505**
Title of Program :	**Lists In prolog**
Objective :	**1. Write a program to manipulate list in prolog Finding Length of Lists** **2. Write a program to manipulate list in prolog Concatenate Two Lists.** **3. Write a program Breadth First Search using Prolog** **4. Write a program To Design Depth First Search using Prolog**

1. Finding Length of Lists

Source Code:

```
/* List manipulation */
/* Finding length of the list */

list_len([],0).
list_len([_|Tail],N):-list_len(Tail,N1),N is N1+1.

list_con([],L,L).
list_con([X1|L1],L2,[X1|L3]):-list_con(L1,L2,L3).
```

Output:

```
| ?- ['list_len.pl'].
compiling C:/Users/mcamock/Desktop/MCA_1505/AIML/l
C:/Users/mcamock/Desktop/MCA_1505/AIML/list_len.pl

yes
| ?- list_len([1,2,3,4],L).

L = 4

yes
| ?- D
```

2. **Concatenate Two Lists.**

Source Code:

```
/* List manipulation */
/* Finding length of the list */

list_len([],0).
list_len([_|Tail],N):-list_len(Tail,N1),N is N1+1.

list_con([],L,L).
list_con([X1|L1],L2,[X1|L3]):-list_con(L1,L2,L3).
```

Output:

```
| ?- ['list_len.pl'].
compiling C:/Users/mcamock/Desktop/MCA_1505/AIML/list_len.pl for byte code.
C:/Users/mcamock/Desktop/MCA_1505/AIML/list_len.pl compiled, 8 lines read -

yes
| ?- list_con([1,2,4],[5,7],Newlist).

Newlist = [1,2,4,5,7]

yes
```

3. Breadth First Search using Prolog

Source Code:

```
/* Breadth First Search Using Prolog */

/* Base Case */
search_bf([Goal|Rest],Goal):-
        goal(Goal),
        write('Goal Found: '),write(Goal),nl.

/* Recursive Case */
search_bf([Current|Rest],Goal):-
        children(Current,Children),
        append(Rest,Children,NewAgenda),
        search_bf(NewAgenda,Goal).

/* Define Graph */
children(a,[b,c]).
children(b,[d,e]).
children(c,[]).
children(d,[]).
children(e,[]).
goal(e).
```

Output:

```
| ?- ['search_bf.pl'].
compiling C:/Users/mcamock/Desktop/MCA_1505/AIML/search_bf.pl for byte code...
C:/Users/mcamock/Desktop/MCA_1505/AIML/search_bf.pl:4-6: warning: singleton variables [Res
C:/Users/mcamock/Desktop/MCA_1505/AIML/search_bf.pl compiled, 19 lines read - 1761 bytes w

yes
| ?- search_bf([a],Goal).
Goal Found: e

Goal = e ?

yes
```

4. **Depth First Search using Prolog**

Source Code:

```
/* Define Graph */
children(a,[b,c]).
children(b,[d,e]).
children(c,[]).
children(d,[]).
children(e,[]).
goal(e).
goal(f).

/*Depth first Search*/
search_df(Start):-
        dfs([Start],Path),
        write('Goal Found: '),write(Path),nl.

/*Base Case */
dfs([Node|RestPath],[Node|RestPath]):-
        goal(Node).

/* Recursive Case */
dfs([Node|RestPath],FinalPath):-
        children(Node,Children),
        member(Child,Children),
```

```
\+ member(Child,[Node|RestPath]),
dfs([Child,Node|RestPath],FinalPath).
```

Output:

```
| ?- ['search_df.pl'].
compiling C:/Users/mcamock/Desktop/MCA_1505/AIML/search_df
C:/Users/mcamock/Desktop/MCA_1505/AIML/search_df.pl compile

yes
| ?- search_df(a).
Goal Found: [e,b,a]

true ?

yes
| ?-
```