CERTIFICATE

This is to certify that Mr.Patel Joy Jayesh Enrollment. No.

19BEIT30033 of Information Technology has satisfactory completed the

Course in : Artificial Intelligence (CT-601N) at LDRP INSTITUTE OF

TECHNOLOGY AND RESEARCH, ***Gandhinagar***

Date of Submission: April 2022

Concern Faculty: Riddhi Patel

Head of Department: Dr. Mehul Barot

KADI SARVA VISHWAVIDYALAYA

LDRP Institute of Technology and Research, Gandhinagar

Practical Index

Subject Code: CT601-N

Subject Title: Artificial Intelligence

Semester: 6th

No	Name of Experiment
1	To study of facts, objects, Predicates, Variables, Rules and Unification in PROLOG.
2	Write a program to implement "cut" and "fail" predicate in PROLOG. Program1: Implement program to find Minimum and Maximum number without using cut and fail predicates in prolog. Program2: Program to find minimum and maximum using cut and fail predicate.
3	Write a program to implement arithmetic operators , simple input/output and compound goals in PROLOG. Program 1: Program to implement arithmetic operator. Program 2: Program to implement Tower of Hanoi. Program 3: Program to find grade. Program4: Program to implement any program which you have implement in c/c++/Java.
4	Write a program to implement recursion in PROLOG. Program 1: Program to mplement factorial. Program 2: Program to implement factorial using cut and fail predicates.
5	Write a program to implement list in PROLOG. Program 1: Prolog program to add an element to the beginning of a list. Program 2: Prolog program Define the relation last (item, list) so that item is the last element of the list using concatenate.
6	Write a program to implement string operations in PROLOG. Implement string operations like substring, String position, palindrome etc.)
7	Write a prolog program to maintain family tree.
8	Write a program to implement BFS (for 8 puzzle problem or Water Jug problem or any AI search problem).
9	Write a program to implement DFS (for 8 puzzle problem or Water Jug problem or any AI search problem).
10	Write a program to Implement A* Algorithm.
11	Write a program to solve travelling salesman problem using Prolog.
12	Study of dynamic database in PROLOG.

PRACTICAL: 1

AIM: To study of facts, Objects, Predicators, Variables, Rules and Unification in PROLOG.

19BEIT30033

AI

DELUXE PAGE NO. : DATE :

Practical 1.

Aim: To study of facts, objects, predicates, variables, Rules and Unification in Prolog

→ Full form of Prolog is Programming Logic. It is different from Java, Python, C, C++.

→ In Prolog Programming, the programmer might begin by telling the computer one fact. So prolog consists of facts & rules.

→ Prolog is object oriented language (OOPS concept) Prolog program is a collection of data or facts and the relationship among those facts.
Ex: The program is database.

☆ Facts, Objects, Relations:

→ Joy is a friend of Nirav.
Here Joy & Nirav are objects and friend is a relation between them.

1) Facts:
→ Facts can be describe as a symbolic Relationship.
Ex: Bob is a student.
student(Bob)

Above factual expression is called clauses.

2) Objects:
→ Object is the name of the element of certain type.

DATE :

→ Object type could be as follow:
1) char: single character
2) Integer: integer
3) Real: floating point number
4) string: character sequence
5) symbol: character sequence of letters, numbers and unders

2) Relation:

→ A Relation is a name that defines the way in which a collection of objects.
→ Bob is a student. Here student is a relation to Bob.

3) Predicates:

→ Car is blue.
is (Car blue)
→ above statement is not fact untill dot(.) is not written at the end. In the prolog(.) dot is known as period.
→ So without period, expression is known as predicates.

★ Parts of Prolog:

1) Editor: In the editor, we write queries/code

2) Dialog: In the dialog part, we write the code/query with the help of a bound, free and anonymous variables.

3) Message: Message part give system message.

4) Trace: Trace part is used when we debug the code.

★ Structure of Prolog:

1) Domains:

→ In the domain part we declare the object and variables. There are different type of variables like symbol, string, integer, character, file, real.

→ // defining objects, variables.

2) Predicates:

→ In the predicates, we define the relation among the objects that we defined in the domains part.

→ // defining a relation function

3) Clauses:

→ In the clauses, we define the facts.

→ // defining facts, rules.

★ Variables:

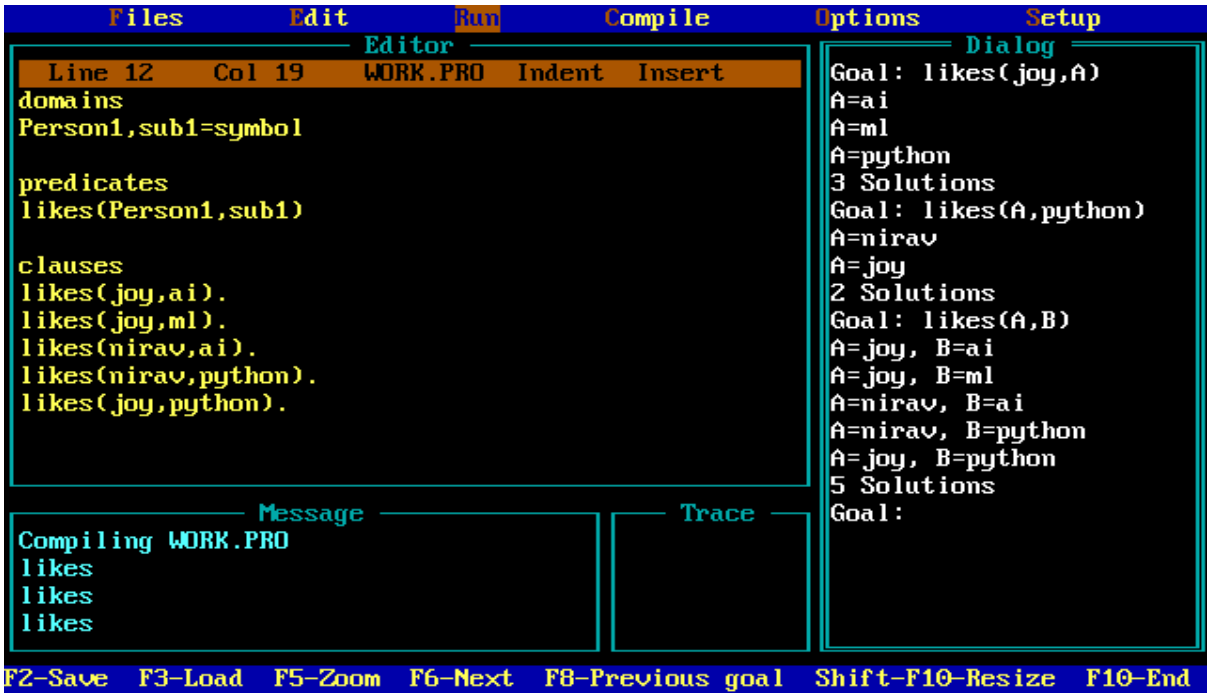
→ There are 3 types of variables:

1) Bound
2) Free
3) Anonymous.

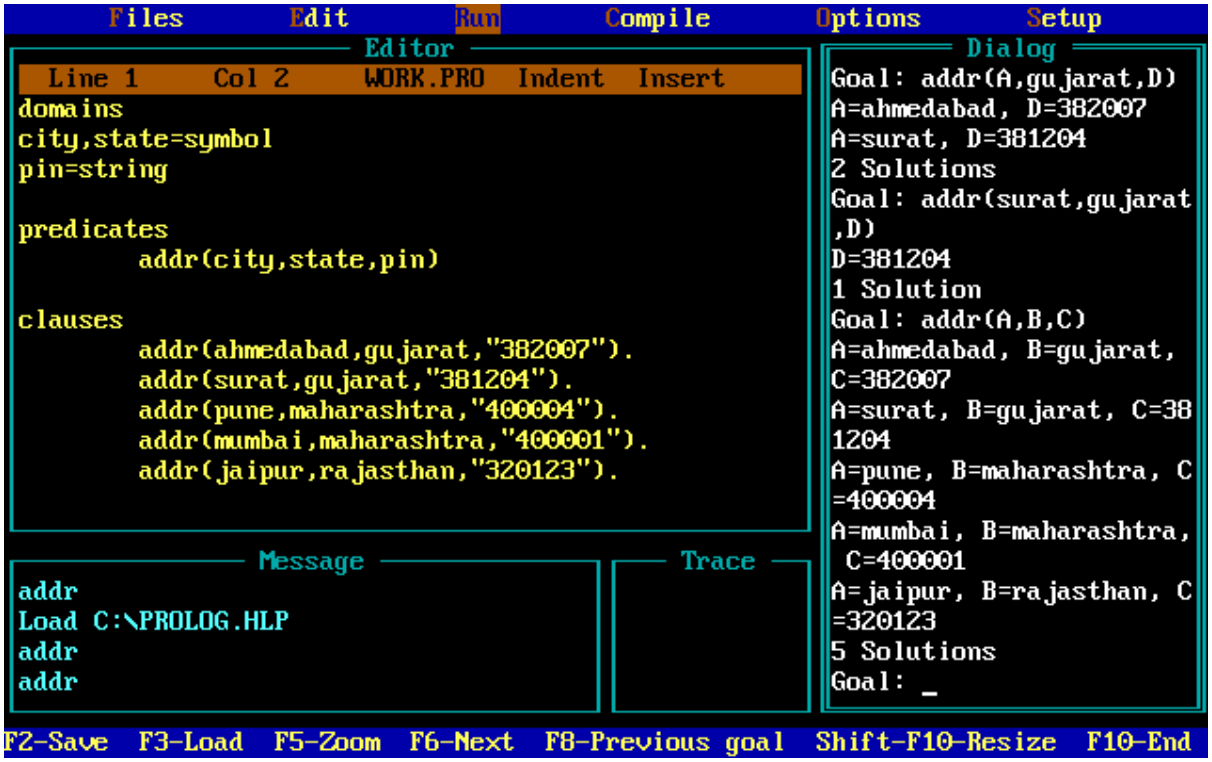
Bound: Means value is assigned to variables.

Free: These variables which are not initialized as defined at particular time is called free variables.

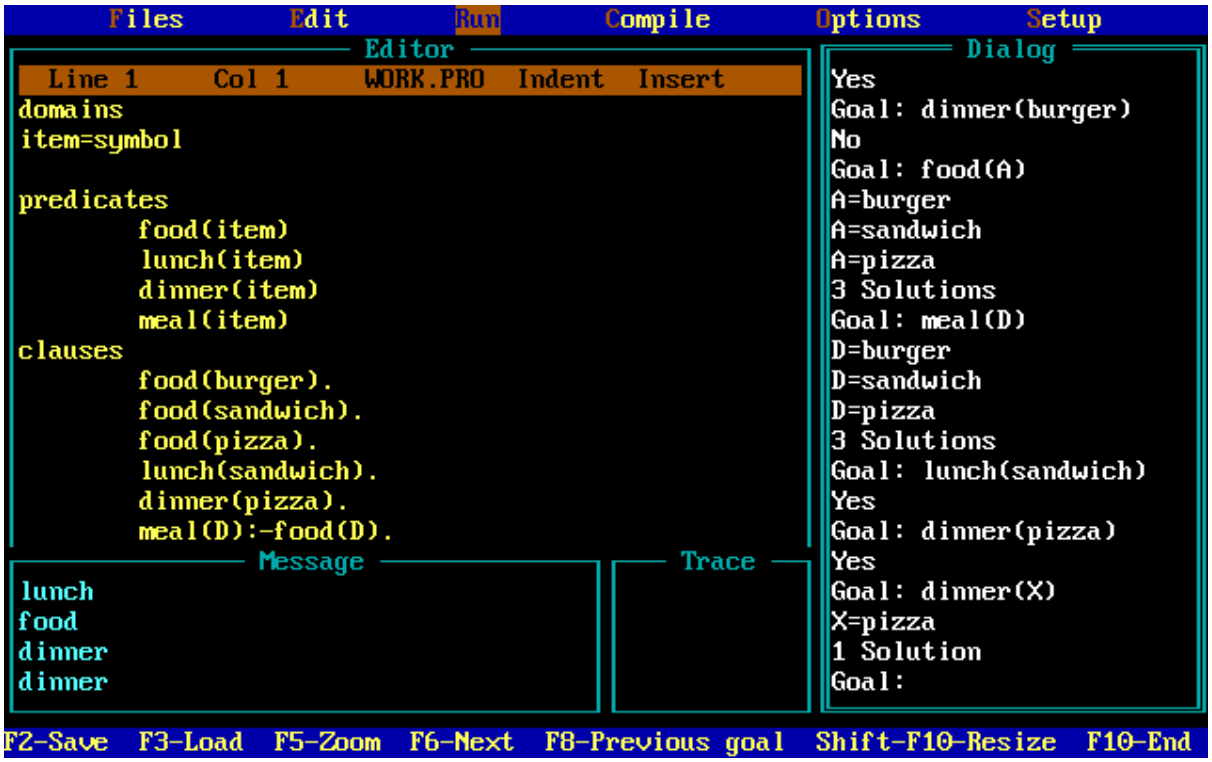
Example: 1



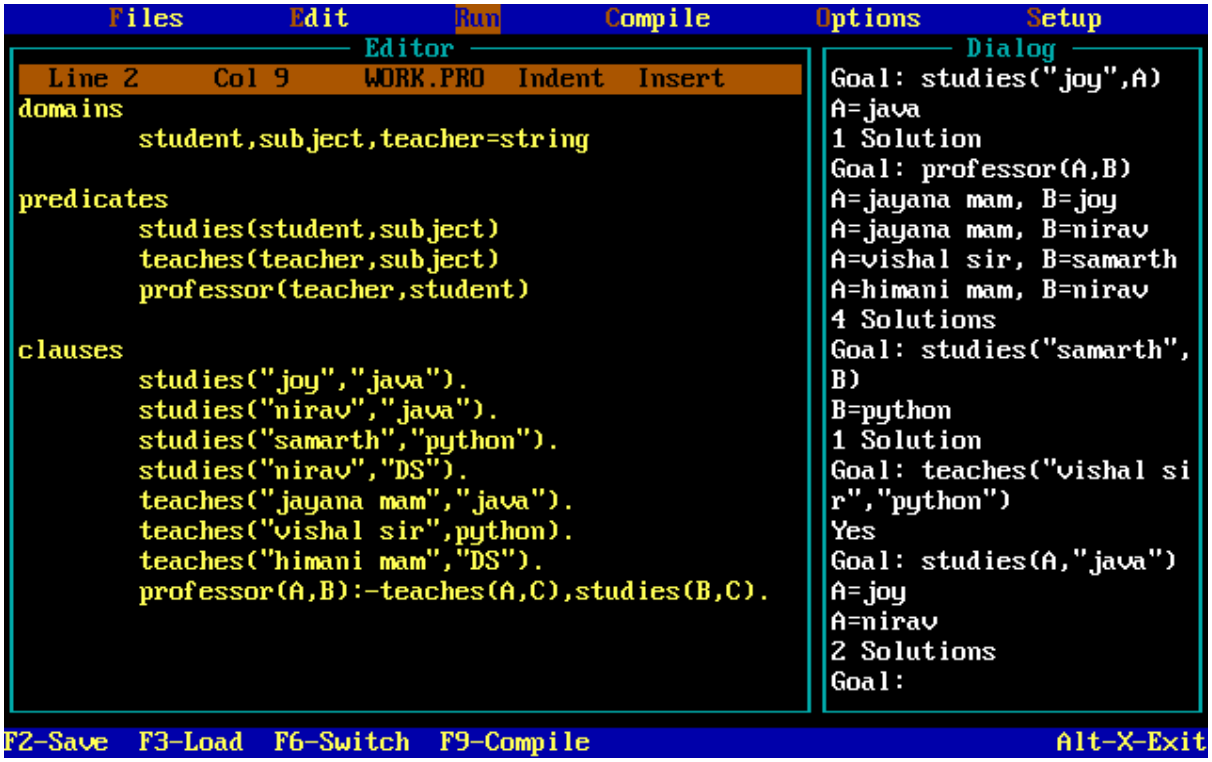
Example 2:



Example 3:



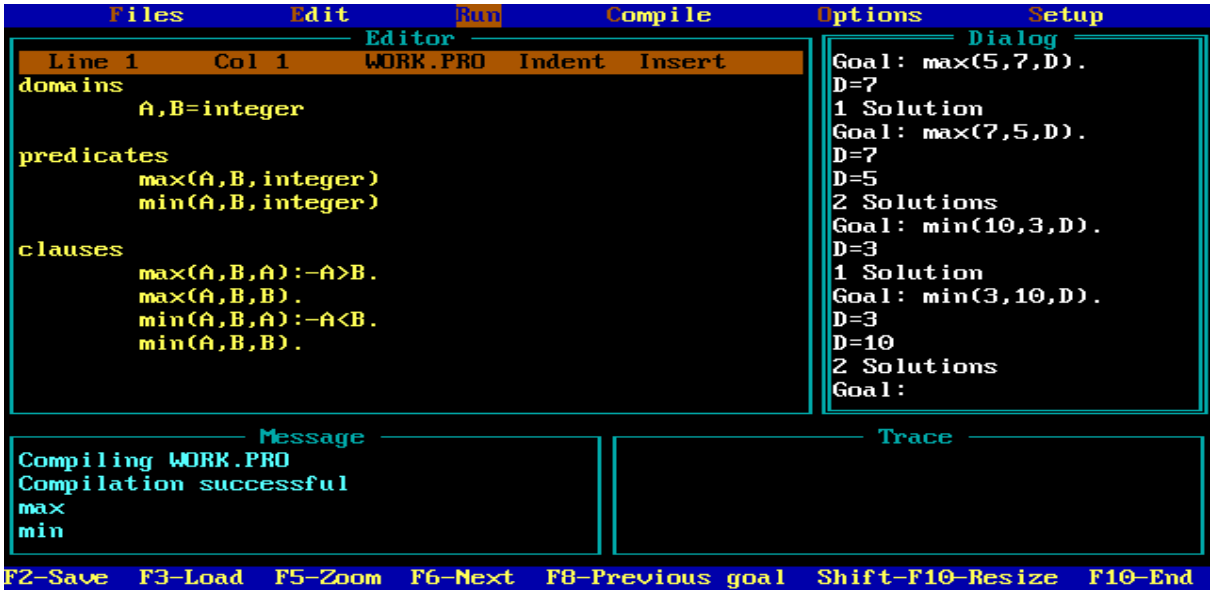
Example 4:



PRACTICAL: 2

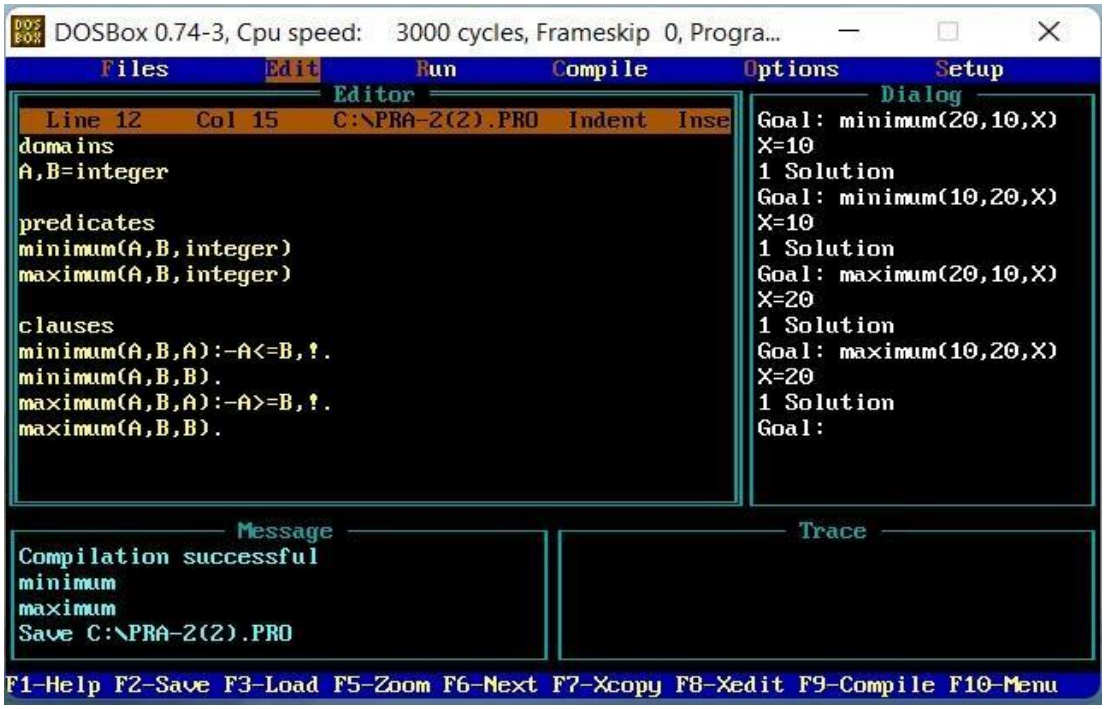
AIM: Write a program to implement “cut” and “fail” predicate in PROLOG.

Program 1: Implement program to find Maximum and Minimum number without using cut and fail predicates in prolog.

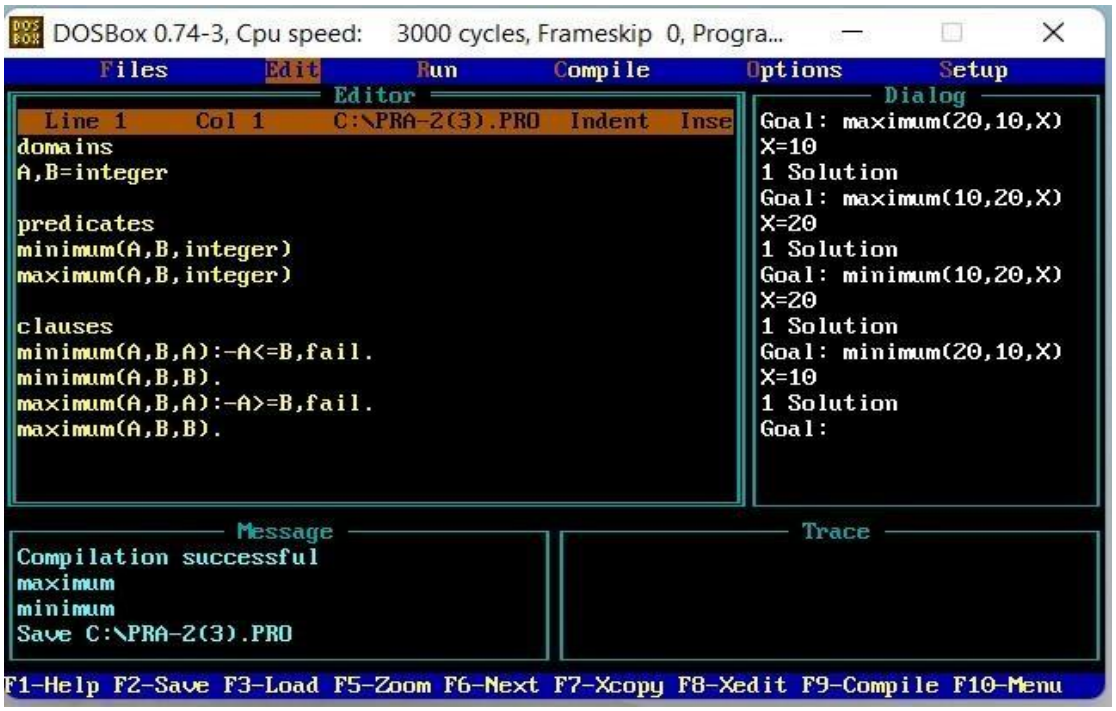


Program 2: Implement program to find Maximum and Minimum number using cut and fail predicates in prolog.

Cut:



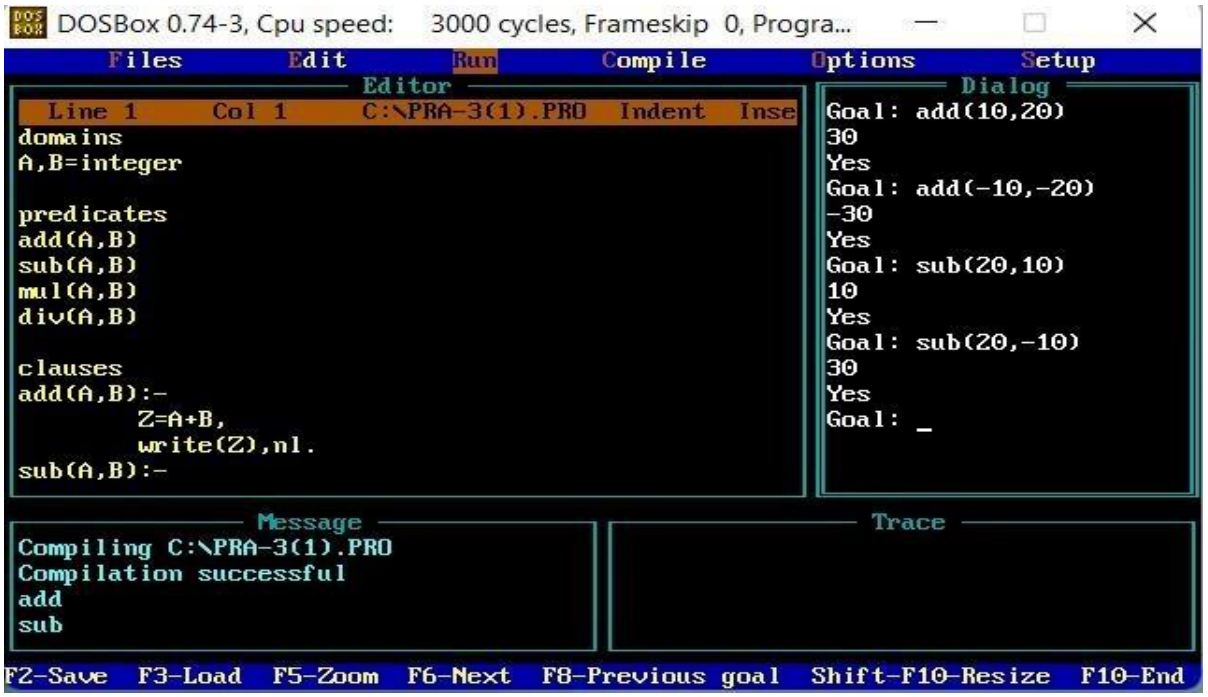
Fail:

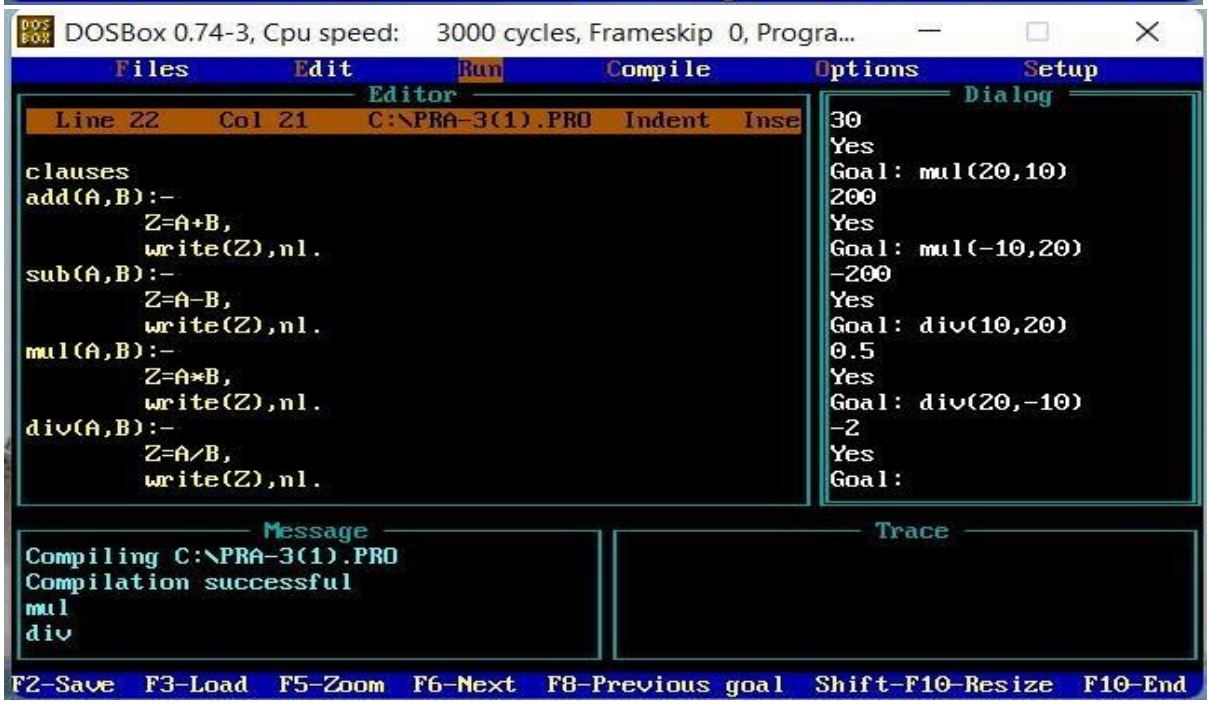


PRACTICAL: 3

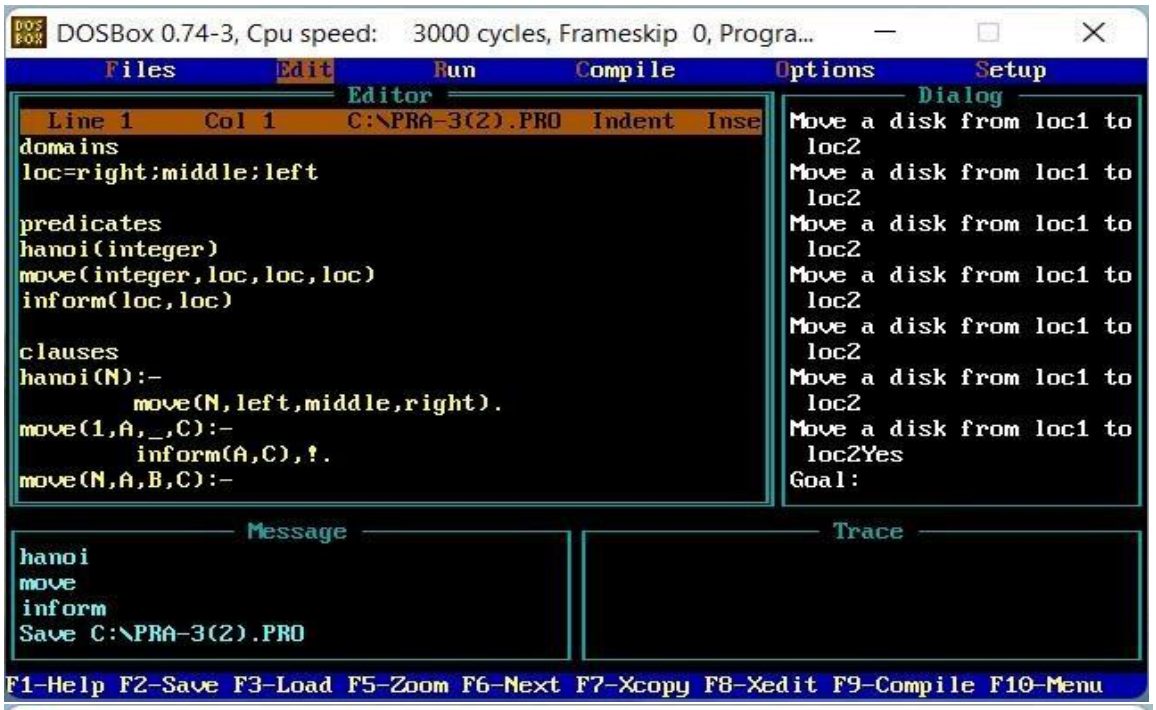
AIM: Write a program to implement arithmetic operators, simple input/output and compound goals in PROLOG.

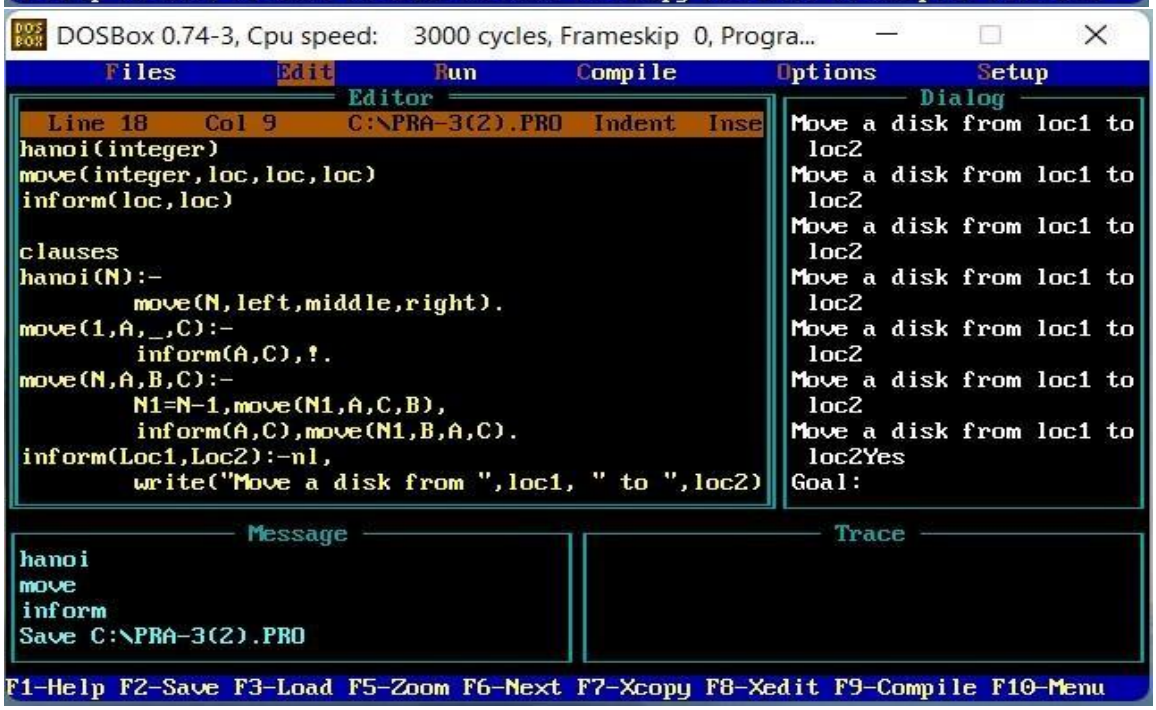
Program 1: Program to implement arithmetic operators.



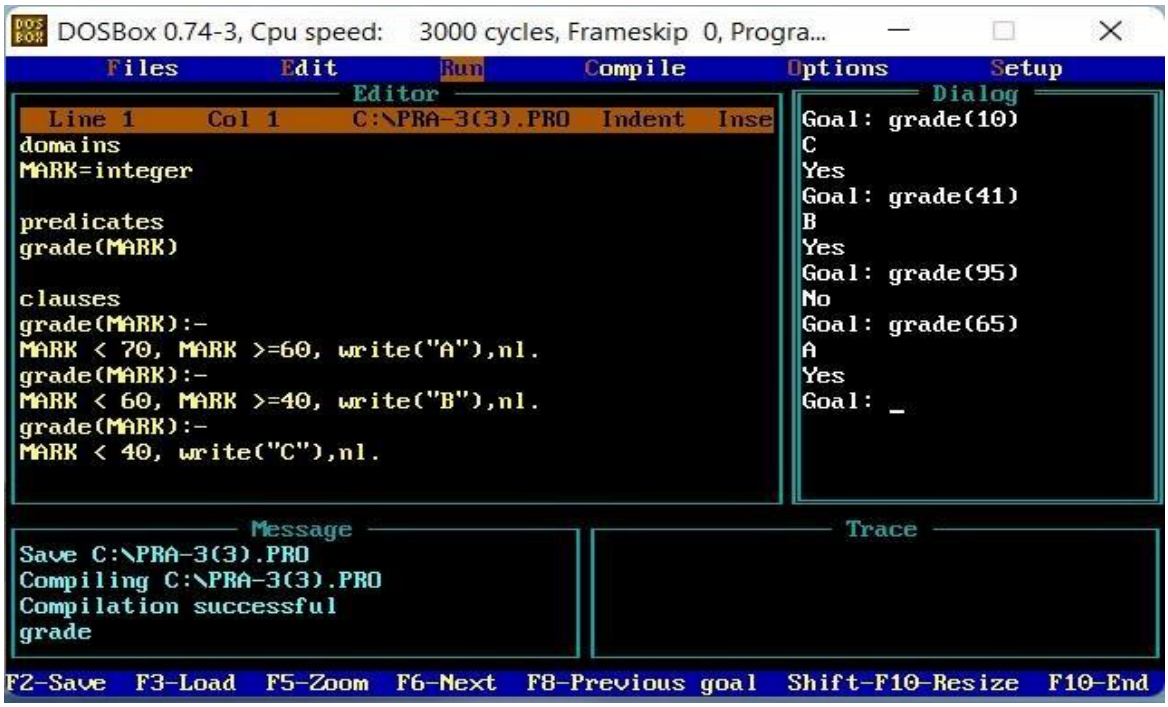


Program 2: Program to implement tower of hanoi.

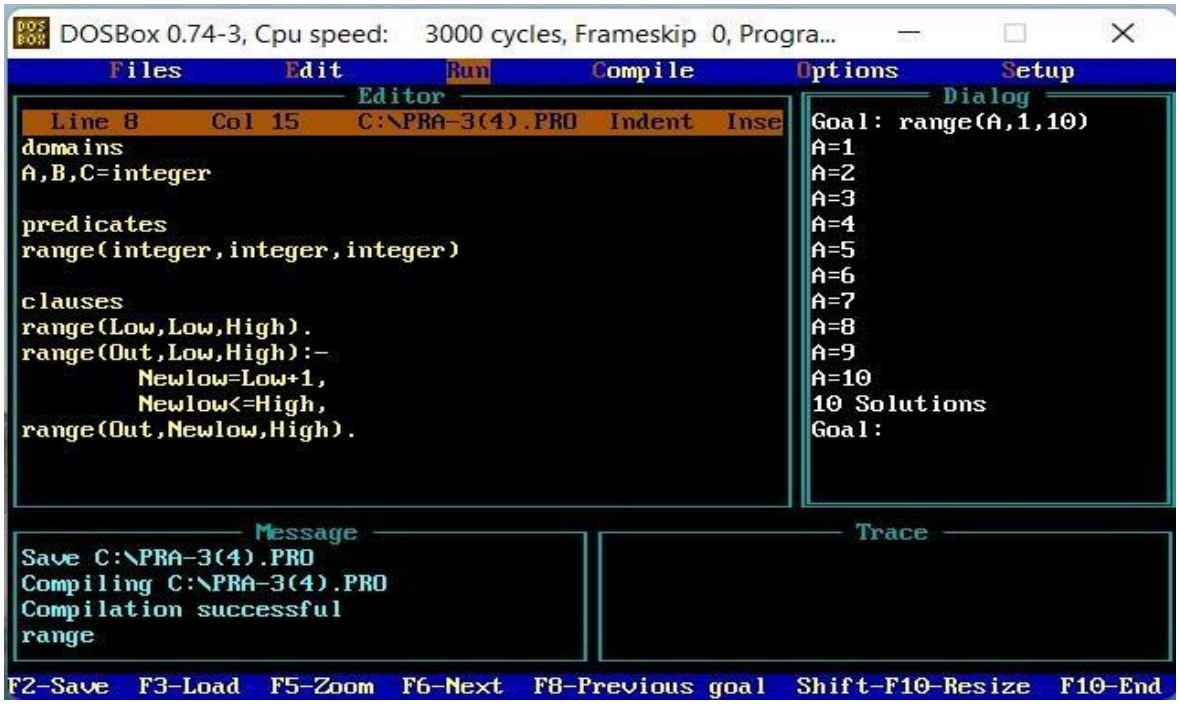




Program 3: Program to find grade.



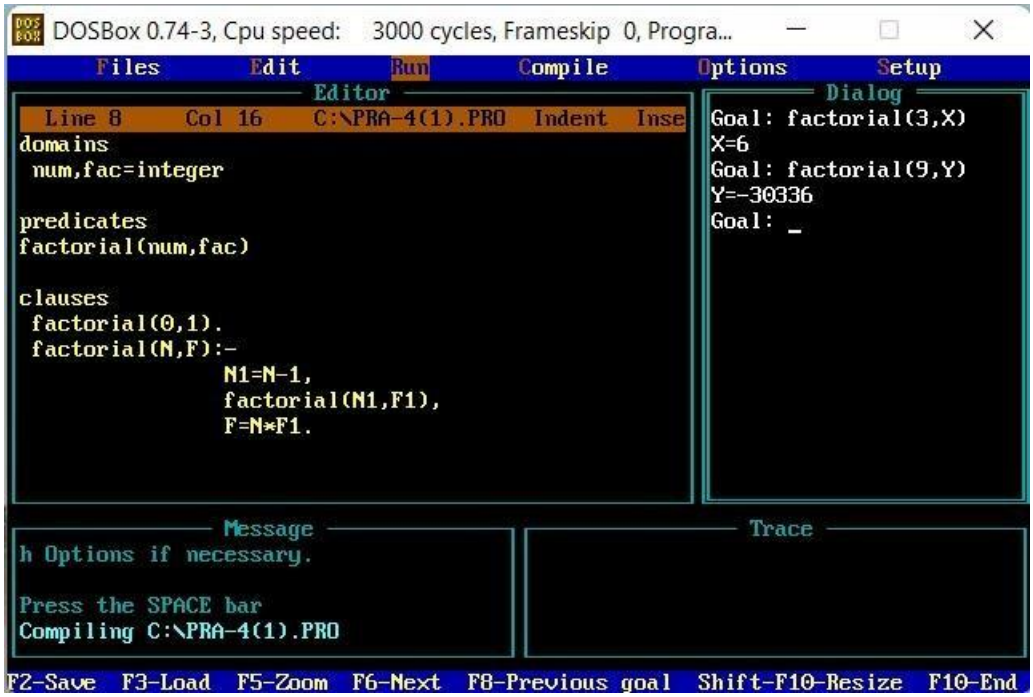
Program 4: Program to print the values between the given range.



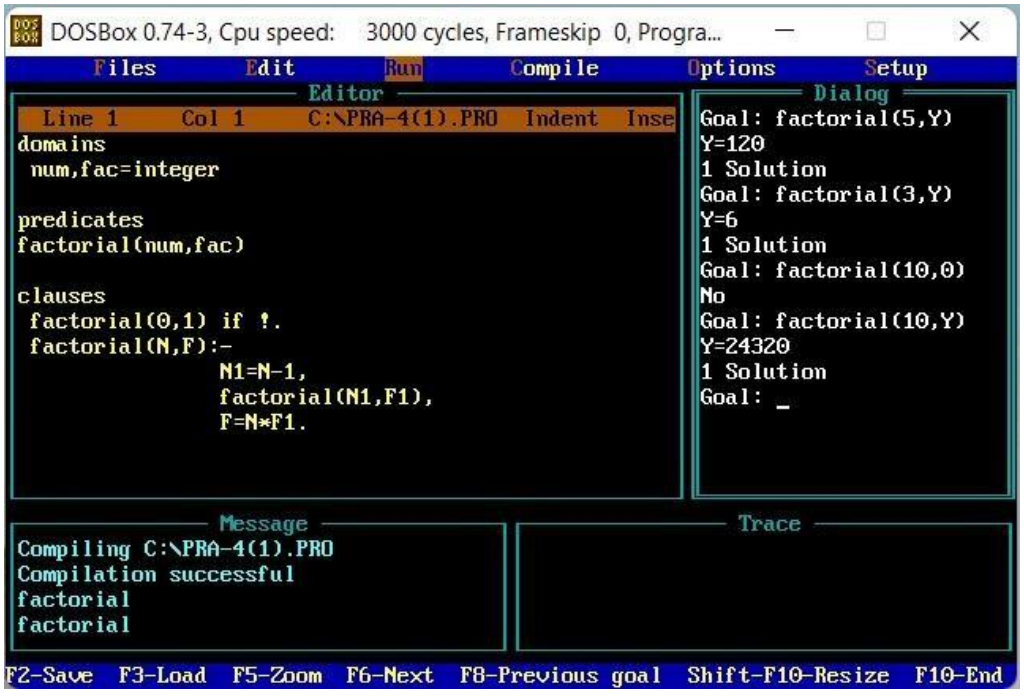
PRACTICAL: 4

AIM: Write a program to implement recursion in PROLOG.

Program 1: Program to implement factorial.



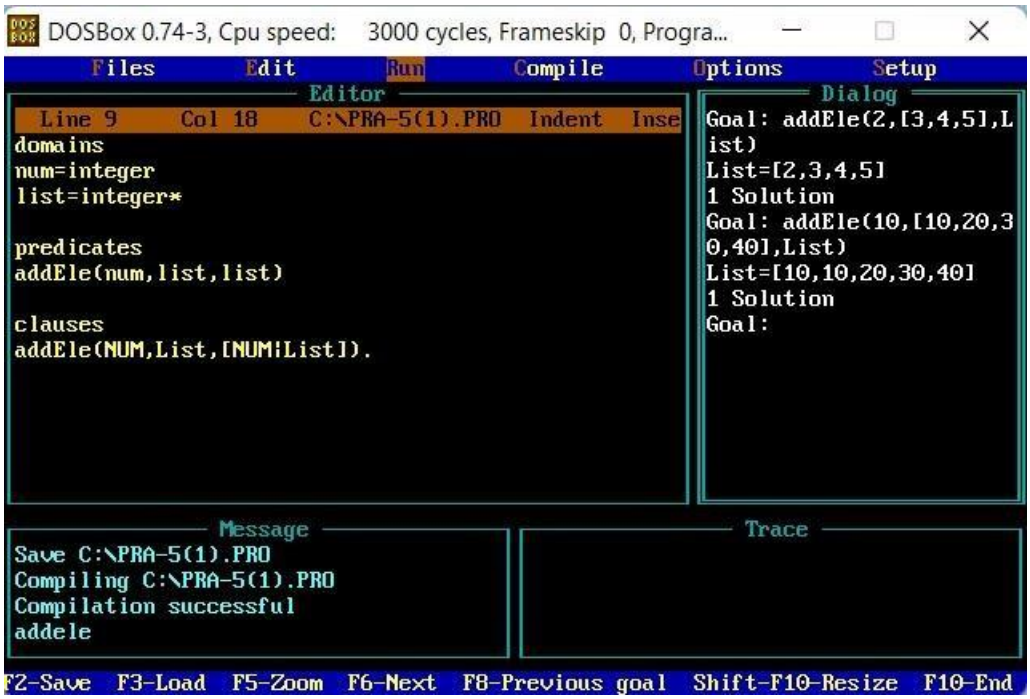
Program 2: Program to implement factorial using CUT and FAIL predicates.



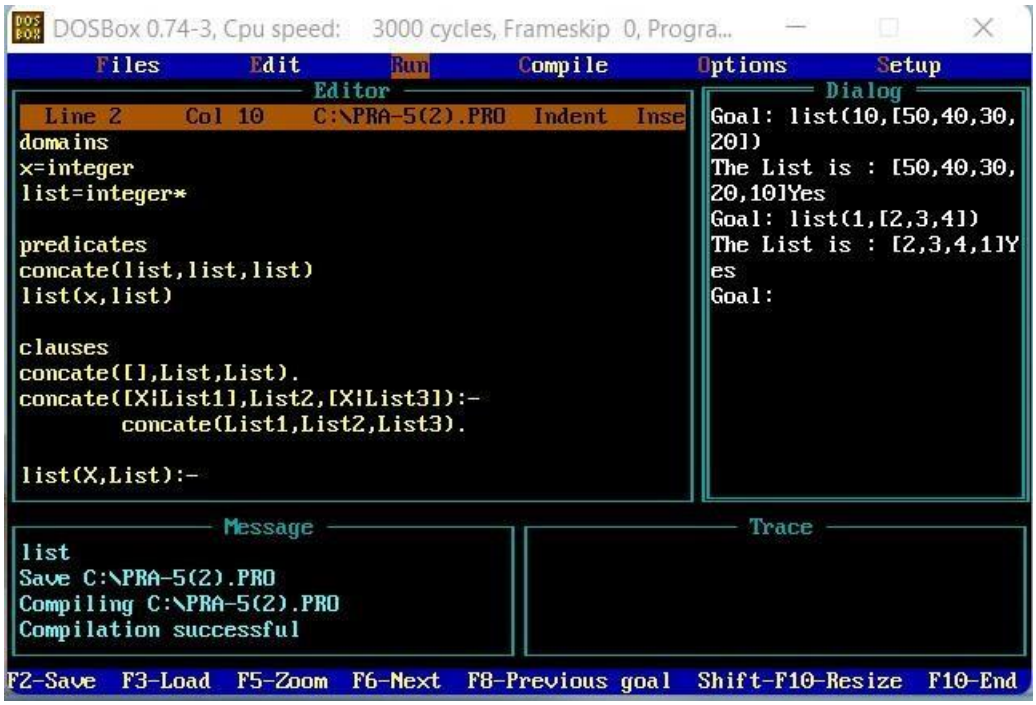
PRACTICAL: 5

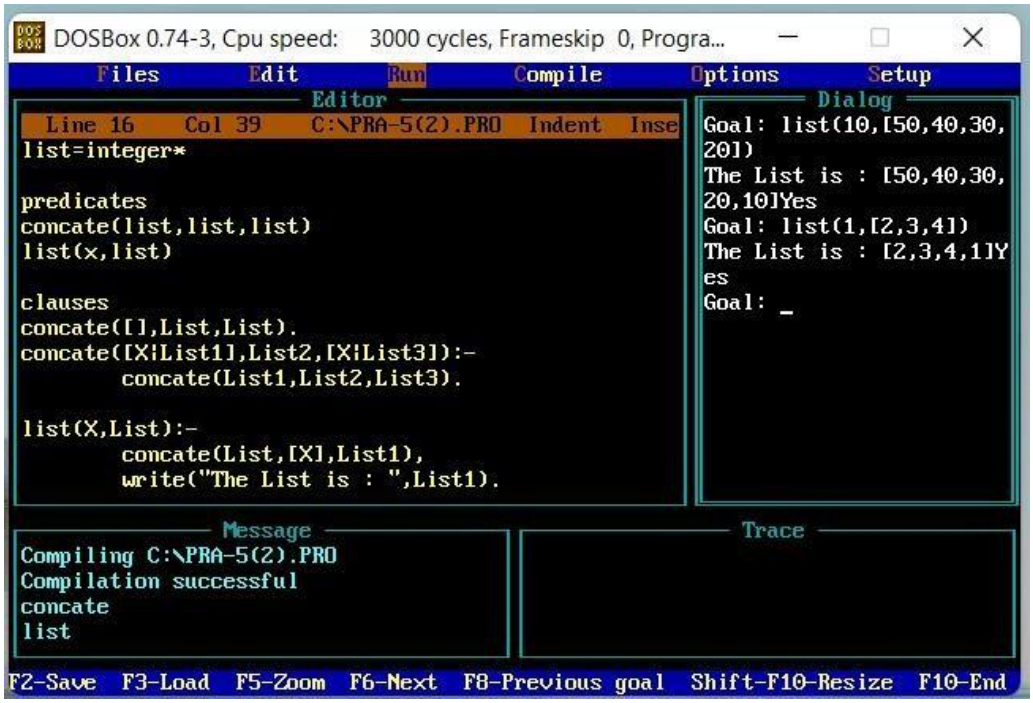
AIM: Write a program to implement Lists in PROLOG.

Program 1: Program to implement add an element to the List in PROLOG.



Program 2: Program to define the relation list(item,list) so that the item is the last element of the list using concatenate



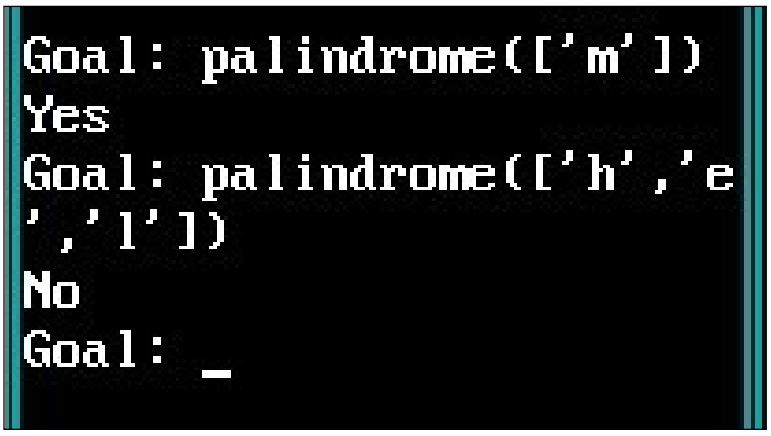

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Files Edit Run Compile Options Setup
Editor
Line 16 Col 39 C:\PRA-5(2).PRO Indent Inse
list=integer*
predicates
concate(list,list,list)
list(x,list)
clauses
concate([],List,List).
concate([X|List1],List2,[X|List3]):-
concate(List1,List2,List3).
list(X,List):-
concate(List,[X],List1),
write("The List is : ",List1).
Dialog
Goal: list(10,[50,40,30,
20])
The List is : [50,40,30,
20,10]Yes
Goal: list(1,[2,3,4])
The List is : [2,3,4,1]Y
es
Goal:
Message
Compiling C:\PRA-5(2).PRO
Compilation successful
concate
list
Trace
F2-Save F3-Load F5-Zoom F6-Next F8-Previous goal Shift-F10-Resize F10-End

PRACTICAL : 6

Program (1): Code for Prolog program to check whether a given word is a palindrome or not in Artificial Intelligence.

```
domains
x = charl = char*

predicates
palindrome(l)
reverse(l,l)
concatenate(l,l, l)

clauses
concatenate([],List,List).

concatenate([X|List1],List2,[X|List 3]) :-
concatenate(List1,List2,List3). reverse([],[]). reverse([X|Tail],List) :- reverse(Tail,Tail1),
concatenate(Tail1,[X],Li st).

palindrome(List):- reverse(List,List).
```

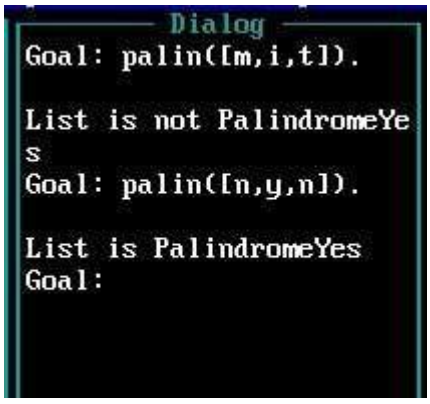
Output:-



Program (2): Code for Prolog program to check whether a given list is palindrome or not in Artificial Intelligence

```
domains list=symbol*

predicates
palin(lis t)
findrev(list,list,li st) compare(list,list)

clauses palin(List1):-
findrev(List1,[],List2),
compare(List1,List2).
findrev([],List1,List1).
findrev([X|Tail],List1,List2):-findrev(Tail,[X|List1],List2).

compare([],[]):-
write("\nList is Palindrome").

compare([X|List1],[X|List2]):-
compare(List1,List2).

compare([X|List1],[Y|List2]):- write("\nList is not Palindrome").
```

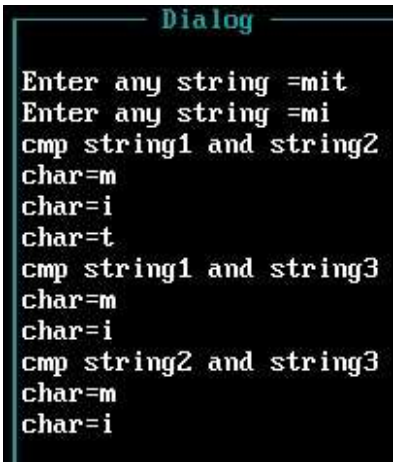
OUTPUT



Program (3): Code for Prolog program to compare characters, strings and also reverse string in Artificial Intelligence

```
domains
strlist =string* predicates
start createlist(integer,strlist,strlist,strlist) strcmp(string,string) charcmp(char,string) reverse(strlist,strlist,strlist)

goal clearwindow, start.

clauses
start: createlist(3,[],Newlist,List 1), reverse(List1,[],List2),List2 = [Str1 |Tail], Tail = [Str2| Tail1],Tail1=[Str3| Str4],
write("cmp string1 and string2"),nl, strcmp(Str1,Str2),write("cmp string1 and string3"),nl, strcmp(Str1,Str3), write("cmp string2 and string3"),nl, strcmp(Str2,Str3).

createlist(Num,Oldlist,Newlist,List1):- Num > 0, write("Enter any string="), readln(Str), Newlist = [Str | Oldlist], NN = Num -1, createlist(NN,Newlist,List2,List1).

createlist(_,Oldlist,_,List1):- List1 = Oldlist.strcmp(Str1,Str2):- frontchar(Str1,Ch1,Rest1), charcmp(Ch1,Str2), Rest1<>"",strcmp(Rest1,Str2).
strcmp(Str1,Str2). charcmp(Ch1,Str2):- frontchar(Str2,Ch2,Rest2), Ch1<>Ch2, Rest2<>"", charcmp(Ch1,Rest 2).

charcmp(Ch1,Str2):- frontchar(Str2,Ch2,Rest2), Ch1=Ch2,write("char=",Ch2),nl.

charcmp(Ch1,Str2). reverse([],Inputlist,Inputlist).reverse([Head|Tail],List1,List2):- reverse(Tail,[Head | List1],List2).
```

Output:-



PRACTICAL : 7.

Program (1): Code for Prolog program for family hierarchy in Artificial Intelligence.

```
predicates
    male(symbol).
    female(symbol).
    father(symbol,symbol).
    husband(symbol,symbol).
    brother(symbol,symbol).
    sister(symbol,symbol).
    listbrothers(symbol).
    listsisters(symbol).
    mother(symbol,symbol).
    grandfather(symbol).
    grandmother(symbol).
    uncle(symbol).
    aunt(symbol).
    cousin(symbol).
    listgrandsons(symbol).
    listgranddaughters(symbol).
    printmenu.
    action(integer).
    repeat.

clauses
        male(dashrath).
        male(ram).
        male(laxman).
        male(bharat).
        male(luv).
        male(kush).
        male(son_of_laxman).

        female(kaushalya).
        female(sita).
        female(urmila).
        female(daughter_of_dashrath).

        father(dashrath,ram).
        father(dashrath,laxman).
        father(dashrath,bharat).
        father(ram,luv).
```

```
father(ram,kush).
father(laxman,son_of_laxman).
father(dashrath,daughter_of_dashrath).

husband(dashrath,kaushalya).
husband(ram,sita).
husband(laxman,urmila).

mother(X,Y):- husband(Z,X),father(Z,Y).
brother(X,Y):- father(Z,X),father(Z,Y),X<>Y,male(X).
sister(X,Y):- father(Z,X),father(Z,Y),X<>Y,female(X).
listbrothers(X) :-brother(Z,X),write(Z).
listsisters(X):-sister(Z,X),write(Z).
grandfather(X):-father(Y,Z), father(Z,X),write(Y, " is the grandfather of ",X,"\n").
grandmother(X):- husband(Z,X),father(Z,V),father(V,Y),write(Y, " is the grandmother of ",X,"\n").
listgrandsons(X):- father(X,Z),father(Z,Y),male(Y),write(Y,"\n"), fail. listgrandsons(X):-
husband(Y,X),father(Y,V),father(V,Z),male(Z),write(Z,"\n"), fail.
listgranddaughters(X):- father(X,Z), father(Z,Y),female(Y),write(Y,"\n"), fail.
listgranddaughters(X):- husband(Y,X), father(Y,V),father(V,Z),female(Z),write(Z,"\n"), fail.
uncle(X):-brother(Z,Y),father(Z,X),male(Y),write(Y,"\n"), fail.
aunt(X):-husband(Z,Y),brother(Z,V),father(V,X),write(Y,"\n"), fail.
cousin(X):-father(Z,X),father(V,Y),Z<>V,brother(V,Z),write(Y,"\n").
repeat.

repeat:- repeat.

    action(1):-     write("\nEnter name of person whose father is to be found: "), readln(X),write("\n"),
    write("Father of ",X,"is:"), father(Z,X), write(Z,"\n"), fail.

    action(2):-     write("\nEnter name of person whose mother is to be found: "),
    readln(X),write("\n"),write("Mother of ",X," is:"),mother(Z,X), write(Z,"\n"), fail.

    action(3):-     write("\nEnter name of person whose brothers are to be found: "), readln(X),write("\n"),
    write("Brothers of ",X,"are:\n"), listbrothers(X),write("\n"), fail.

    action(4):-     write("\nEnter name of person whose sisters are to be found: "), readln(X),write("\n"),
    write("Sisters of ",X,"are:\n"), listsisters(X),write("\n"), fail.

    action(5):-     write("\nEnter name of person whose grandsons are to be found: "), readln(X),write("\n"),
    write("Grandsons of ",X,"are:\n"), listgrandsons(X),write("\n"), fail.

    action(6):-     write("\nEnter name of person whose granddaughters are to be found: "),
    readln(X),write("\n"), write("Granddaughters of ",X,"are:\n"), listgranddaughters(X),write("\n"), fail.

    action(7):-     write("\nEnter name of person whose uncles are to be found: "), readln(X),write("\n"),
    write("Uncles of ",X,"are:\n"), uncle(X), write("\n"), fail.

    action(8):-     write("\nEnter name of person whose aunties are to be found: "), readln(X),write("\n"),
        write("Aunties of ",X," are:\n"),aunt(X), write("\n"), fail.
```

```
action(9):-     write("\nEnter name of person whose cousins are to be found: "), readln(X),write("\n"),
write("Cousins of ",X,"are:\n"), cousin(X), write("\n"), fail.
action(0). printmenu:-repeat,
    write("\n1. Display Father of?\n"),
    write("2. Display Mother of?\n"),
    write("3. List all brothers of?\n"),
    write("4. List all sisters of?\n"),
    write("5. List all grandson of?\n"),
    write("6. List all granddaughter of?\n"),
    write("7. List all uncles of?\n"),
    write("8. List all aunty of?\n"),
    write("9. list all cousins of?\n"),
    write("0. exit\n"),
    write("Enter your choice :"),
    readInt(Choice), action(Choice), write("\n"),repeat.

goal
 makewindow(1,2,3,"FamilyTree",0,0,25,80), printmenu.
```

Output

```
+-----------------------------Family Tree-----------------------+
¦                                                               ¦
¦1. Display Father of?                                          ¦
¦2. Display Mother of?                                          ¦
¦3. List all brothers of?                                       ¦
¦4. List all sisters of?                                        ¦
¦5. List all grandson of?                                       ¦
¦6. List all granddaughter of?                                  ¦
¦7. List all uncles of?                                         ¦
¦8. List all aunty of?                                          ¦
¦9. list all cousins of?                                        ¦
¦0. exit                                                        ¦
¦Enter your choice : 1                                          ¦
¦                                                               ¦
¦Enter name of person whose father is to be found : ram         ¦
¦                                                               ¦
¦Father of ram is:dashrath                                      ¦
¦                                                               ¦
¦1. Display Father of?                                          ¦
¦2. Display Mother of?                                          ¦
¦3. List all brothers of?                                       ¦
¦4. List all sisters of?                                        ¦
¦5. List all grandson of?                                       ¦
¦6. List all granddaughter of?                                  ¦
¦7. List all uncles of?                                         ¦
¦8. List all aunty of?                                          ¦
¦9. list all cousins of?                                        ¦
¦0. Exit                                                        ¦
                                                                ¦
```

```
Enter your choice : 3

Enter name of person whose brothers are to be found : ram

Brothers of ram are: laxman bharat

1. Display Father of?
2. Display Mother of?
3. List all brothers of?
4. List all sisters of?
5. List all grandson of?
6. List all granddaughter of?
7. List all uncles of?
8. List all aunty of?
9. list all cousins of?
0. exit
Enter your choice : 5

Enter name of person whose grandsons are to be found : dashrath

Grandsons of dashrath are: luv kush son_of_laxman

1. Display Father of?
2. Display Mother of?
3. List all brothers of?
4. List all sisters of?
5. List all grandson of?
6. List all granddaughter of?
7. List all uncles of?
8. List all aunty of?
9. list all cousins of?
0. exit
Enter your choice : 7

Enter name of person whose uncles are to be found : kush

Uncles of kush are:
Laxman
bharat

1. Display Father of?
2. Display Mother of?
3. List all brothers of?
4. List all sisters of?
5. List all grandson of?
6. List all granddaughter of?
7. List all uncles of?
8. List all aunty of?
9. list all cousins of?
0. Exit
Enter your choice :
```

PRACTICAL : 8

Program : Write a program to implement BFS (for AI search problem).

```
domains
        X, H, N, ND=symbol
        P, L, T, Z, Z1, L1, L2, L3, PS, NP, ST, SOL=symbol*

predicates
        solve(L, L) member(X,L) extend(L, L) conc(X, L, L) breadthfirst(L, L) goal(X)


clauses

        solve(start, solution):-/*solution is a state from start to a goal*/ breadthfirst ([[start]], solution).

        breadthfirst([[node|path]| _ ] ,[node|path]):- /*solution is an extension to a goal*/
        /*of one of path*/

        goal(node).

        breadthfirst([path|paths], solution):- extend(path,newpaths), conc(paths,newpaths,path1),
        breadthfirst(path1,solution).

        extend([node|path],newpaths):- bagof([newnode, node|path],(s(node,
        newnode),notmember(newnode,[node|path])), newpaths),!. extend(path, []).
        conc([], L, L).

        conc([X|L1], L2, [X|L3]):- conc(L1, L2, L3).

        member(X, [X|T]).

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

OUTPUT:-

```
goal: solve([a, e], S)
L= [“a”, “b”, ”c”, “d”, ”e”]

goal: solve([a, h],S)
L= [“a”, “b”, ”c”, “d”, ”e”, ”f”, “g”, “h”]
```

PRACTICAL : 9

Program : Write a program to implement DFS (for 8 puzzle problem)

```
domains
        H=integer T=integer*

predicates
        safe(T)
        solution(T)
        permutation(T, T)
        del(H,T,T)
        noattack(H,T,H)

clauses
del(I,[I|L],L). /*to take a position from the permutation oflist*/
del(I,[F|L],[F|L1]):-del(I,L,L1).
permutation([],[]). /*to find the possible positions*/
permutation([H|T],PL):- permutation(T,PT),\ del(H,PL,PT).
solution(Q):- /*final solution is stored in Q*/
permutation([1,2,3,4,5,6,7,8],Q), safe(Q).
safe([]). /*Q is safe such that no queens attack each other*/
safe([Q|others]):-safe(others), noattack(Q,others,1).
noattack(_,[],_)./*to find if the queens are in same row, column or diagonal*/
noattack(Y,[Y1|Ydist],Xdist):- Y1-Y>Xdist, Y- Y1<>Xdist,dist1=Xdist, noattack(Y,Ydist,dist1).
```

OUTPUT:-

```
goal:-solution(Q). Q=["3","8","4","7","1","6",2","5"]
```

PRACTICAL : 10

Program : Write a program to implement A* Algorithm.

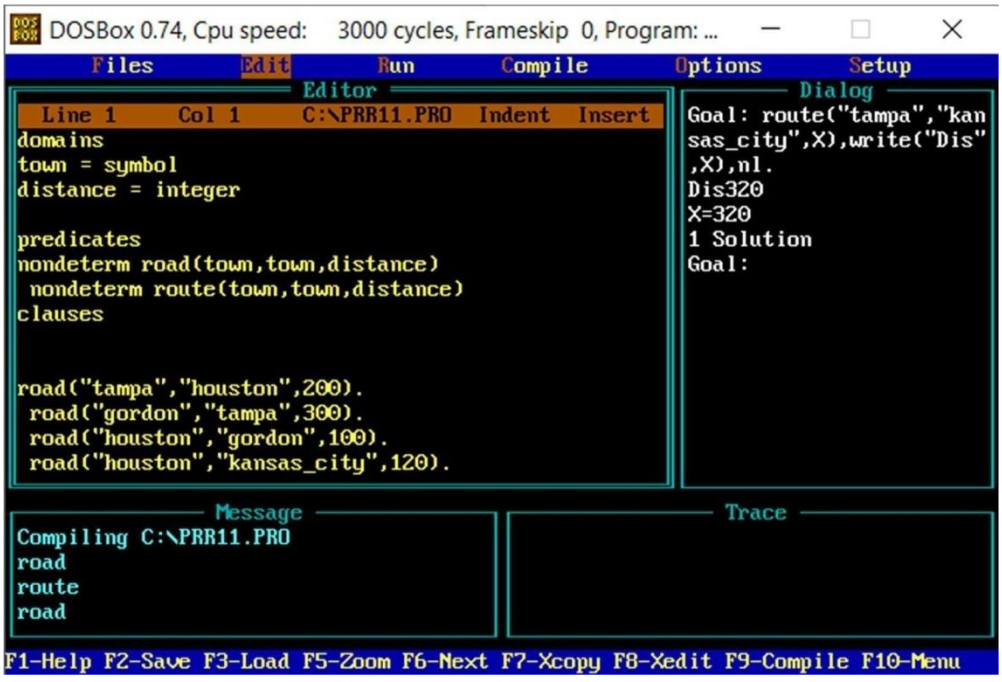
```
%%%
%%%
%%% Nodes have form S#D#F#A
%%%       where S describes the state or

configuration
%%%           D is the depth of the node
%%%           F is the evaluation function value
%%%           A is the ancestor list for the node

:- op(400,yfx,'#'). /* Node builder notation

*/ solve(State,Soln) :- f_function(State,0,F),
              search([State#0#F#[]],S), reverse(S,Soln).
f_function(State,D,F) :- h_function(State,H),
                F is D + H.
search([State#_#_#Soln|_], Soln) :-
goal(State). search([B|R],S) :-
expand(B,Children),
             insert_all(Children,R,Ope
             n), search(Open,S).
insert_all([F|R],Open1,Open3) :- insert
(F,Open1,Open2),
                      insert_all(R,Open2,Open3).
insert_all([],Open,Open).

insert(B,Open,Open) :- repeat_node(B,Open), !
. insert(B,[C|R],[B,C|R]) :- cheaper(B,C), ! .
insert(B,[B1|R],[B1|S]) :- insert(B,R,S),
!. insert(B,[],[B]).
repeat_node(P#_#_#_, [P#_#_#_|_]).

cheaper( _#_#F1#_ , _#_#F2#_ ) :- F1 <

F2.

expand(State#D#_#S,All_My_Childr
   en) :-
   bagof(Child#D1#F#[Move|S],
       (D1 is D+1,
        move(State,Child,Mov
       e),
       f_function(Child,D1,F))
       , All_My_Children).
```

PRACTICAL : 11

Program : Write a program to solve travelling salesman problem using Prolog.

```
domains

        town = symbol
        distance = integer

predicates
        nondeterm road(town,town,distance)
         nondeterm route(town,town,distance)
clauses


        road("tampa","houston",200).
         road("gordon","tampa",300).
         road("houston","gordon",100).
        road("houston","kansas_city",120).
        road("gordon","kansas_city",130).
        route(Town1,Town2,Distance):-
        road(Town1,Town2,Distance).
        route(Town1,Town2,Distance):-
        road(Town1,X,Dist1),
        route(X,Town2,Dist2),
        Distance=Dist1+Dist2, !.
```

OUTPUT:-



PRACTICAL : 12

Program : Study of dynamic database in PROLOG.

```
Predicates
        reading writing delete
        find(integer)
        startup(integer)

Database
        unsortedDatabase(string,integer)sortedDatabase(string)

Clauses
        startup(0).
        startup(Num):- write("Enter String = "),readln(Name), str_len(Name,Len),
        asserta(unsortedDatabase(Name,Len)),TempNum = Num - 1, startup(TempNum).

        writing:- sortedDatabase(Name),write(Name),nl, fail.
        writing.
        find(Index):-
        unsortedDatabase(Name,Index),assertz(sortedDatabase(Name)),retract(unsortedDatabase(Name,Index)),
        find(Index).
        find(Index):-Index = 255.
        find(Index):-TempIndex = Index + 1,find(TempIndex).
        reading:-NumRead= 10, startup(NumRead).

        delete :- retract(sortedDatabase(_)),fail.
        delete.
        Goal
        Clearwindow,makewindow(1,2,3,"String Operations",0,0,25,80),reading,!,find(1),write("\nString In
        Increasing Order Of Their Length Are : \n"),writing,delete.
```

Output :